z/OS

# DFSMS: Managing Catalogs

z/OS

# DFSMS: Managing Catalogs

# Contents

# Figures

# Tables

# About This Book

This publication is intended to help you, a system programmer or storage administrator, build, maintain, and support integrated catalog facility (ICF) catalogs. It also contains the information you need to Virtual Storage Access Method (VSAM) catalogs and OS CVOLs to ICF catalogs.

## Required Product Knowledge

To use this book effectively, you should be familiar with:
- VSAM
- access method services
- storage administration

## Referenced Publications

The following publications are referenced in this book:

| Publication Title | Order Number |
| --- | --- |
| z/OS DFSMS Access Method Services | SC26-7394 |
| z/OS DFSMS: Using Data Sets | SC26-7410 |

## Accessing z/OS DFSMS Books on the Internet

In addition to making softcopy books available on CD-ROM, IBM provides access to unlicensed z/OS softcopy books on the Internet. To find z/OS books on the Internet, first go to the z/OS home page: http://www.ibm.com/servers/eserver/zseries/zos

From this Web site, you can link directly to the z/OS softcopy books by selecting the Library icon. You can also link to IBM Direct to order hardcopy books.

## Using LookAt to look up message explanations

LookAt is an online facility that allows you to look up explanations for z/OS messages, system abends, and some codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can access LookAt from the Internet at:

`http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html`

or from anywhere in z/OS where you can access a TSO command line (for example, TSO prompt, ISPF, z/OS UNIX System Services running OMVS).

To find a message explanation on the Internet, go to the LookAt Web site and simply enter the message identifier (for example, `IAT1836` or `IAT*`). You can select a specific release to narrow your search. You can also download code from the *z/OS Collection*, SK3T-4269 and the LookAt Web site so you can access LookAt from a PalmPilot (Palm VIIx suggested).

To use LookAt as a TSO command, you must have LookAt installed on your host system. You can obtain the LookAt code for TSO from a disk on your *z/OS Collection*, SK3T-4269 or from the LookAt Web site. To obtain the code from the LookAt Web site, do the following:

1. Go to http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookat.html.
2. Click the **News** button.
3. Scroll to **Download LookAt Code for TSO and VM.**
4. Click the ftp link, which will take you to a list of operating systems. Select the appropriate operating system. Then select the appropriate release.
5. Find the **lookat.me** file and follow its detailed instructions.

To find a message explanation from a TSO command line, simply enter: **lookat** *message-id*. LookAt will display the message explanation for the message requested.

**Note:** Some messages have information in more than one book. For example, IEC192I has routing and descriptor codes listed in *z/OS MVS Routing and Descriptor Codes*. For such messages, LookAt prompts you to choose which book to open.

## Accessing Licensed Books on the Web

z/OS licensed documentation in PDF format is available on the Internet at the IBM Resource Link Web site at:

`http://www.ibm.com/servers/resourcelink`

Licensed books are available only to customers with a z/OS license. Access to these books requires an IBM Resource Link Web userid and password, and a key code. With your z/OS order you received a memo that includes this key code.

To obtain your IBM Resource Link Web userid and password log on to:

`http://www.ibm.com/servers/resourcelink`

To register for access to the z/OS licensed books:

1. Log on to Resource Link using your Resource Link user ID and password.
2. Click on **User Profiles** located on the left-hand navigation bar.
3. Click on **Access Profile.**
4. Click on **Request Access to Licensed books.**
5. Supply your key code where requested and click on the **Submit** button.

If you supplied the correct key code you will receive confirmation that your request is being processed. After your request is processed, you will receive an e-mail confirmation.

**Note:** You cannot access the z/OS licensed books unless you have registered for access to them and received an e-mail confirmation informing you that your request has been processed.

To access the licensed books:

1. Log on to Resource Link using your Resource Link userid and password.
2. Click on **Library**.
3. Click on **zSeries**.

4. Click on **Software**.
5. Click on **z/OS**.
6. Access the licensed book by selecting the appropriate element.

## How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. If you have any comments about this book or any other DFSMS documentation:

- Send your comments by e-mail to:
  - IBMLink™ from US: starpubs@us.ibm.com
  - IBMLink from Canada: STARPUBS at TORIBM
  - IBM Mail Exchange: USIB3VVD at IBMMAIL
  - Internet: starpubs@us.ibm.com

  Be sure to include the name of the book, the part number of the book, version and product name, and if applicable, the specific location of the text you are commenting on (for example, a page number or a table number).

- Fill out one of the forms at the back of this book and return it by mail or by giving it to an IBM representative. If the form has been removed, address your comments to IBM Corporation, RCF Processing Department M86/050, 5600 Cottle Road, San Jose, California 95193-0001, U.S.A.

# Summary of Changes

This book contains terminology, maintenance, and editorial changes. A vertical line indicates technical changes or additions to the text and illustrations to the left of the change.

## Summary of Changes for SC26-7409-01 z/OS Version 1 Release 3

This book contains information previously presented in *z/OS Version 1 Release 1 Managing Catalogs* (SC26-7409-00).

The following sections summarize the changes to that information.

## New Information

The changes for this release are designed to help you self-diagnose problems when using catalog functions including:

- The catalog address space is listed in an switched virtual channel (SVC) dump when catalog is actively processing for the dumped job. This improves the first failure data capture when a failure occurs because catalog information is available in an SVC dump in all cases where a catalog is being used by that address space. Therefore, you do not need to re-create the problem in the event that the problem is in the catalog address space.
- Invalid catalog data set names are not allowed. If you use an invalid catalog data set name, an error message is displayed before you run your job. This saves time because you can correct the error before you run a job that will fail because of an invalid catalog data set name.
- Additional information is included in the SYMREC record which decreases the probability that you will have to re-create a problem in order to diagnose it.

## Deleted Information

Chapter 9, ″Converting to Integrated Catalog Facility Catalogs″ was deleted

## Library Changes for z/OS Version 1 Release 3

You might notice changes in the style and structure of some content in this book–for example, headings that are more task-oriented, notes with headings that are specific and clear in their intent, additional index entries for easier information retrieval, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our books.

# Chapter 1. Introduction to the Integrated Catalog Facility

A catalog is a data set that contains information about other data sets. It provides users with the ability to locate a data set by name, without knowing where the data set resides. By cataloging data sets, your users will need to know less about your storage setup. Thus, data can be moved from one device to another, without requiring a change in JCL DD statements which refer to an existing data set.

Cataloging data sets also simplifies backup and recovery procedures. Catalogs are the central information point for VSAM data sets; all VSAM data sets must be cataloged. In addition, all SMS-managed data sets must be cataloged.

DFSMS provides the Integrated Catalog Facility (ICF) catalog. Any type of data set or object can be cataloged in an ICF catalog. Many advanced functions require the use of ICF catalogs, for example, the Storage Management Subsystem.

**Attention:** VSAM Catalogs and OS CVOLs are not supported after January 1, 2000. Any attempt to access them as a catalog will fail.

## Integrated Catalog Facility Catalog Structure

An ICF catalog consists of two separate kinds of data sets: a basic catalog structure (BCS); and a VSAM volume data set (VVDS). The BCS can be considered *the* catalog, whereas the VVDS can be considered an extension of the volume table of contents (VTOC).

The **basic catalog structure** is a VSAM key-sequenced data set. It uses the data set name of entries to store and retrieve data set information. For VSAM data sets, the BCS contains volume, security, ownership, and association information. For non-VSAM data sets, the BCS contains volume, ownership, and association information.

The **VSAM volume data set** is a VSAM entry-sequenced data set. A VVDS resides on every volume which contains a VSAM or SMS-managed data set cataloged in an ICF catalog. It contains the data set characteristics, extent information, and the volume-related information of the VSAM data sets cataloged in the BCS. If you are using the Storage Management Subsystem (SMS), the VVDS also contains data set characteristics and volume-related information for the non-VSAM, SMS-managed data sets on the volume.

The **Volume Table of Contents** and the VTOC index are system data sets which maintain extent and allocation information for a volume. The VTOC is used to find empty space for new allocations and to locate non-VSAM data sets. For all VSAM data sets, and for SMS-managed non-VSAM data sets, the VTOC is used to obtain information not kept in the VVDS.

VVDS records for VSAM data sets are called "VSAM volume records" (VVRs). Those for SMS-managed non-VSAM data sets are called "non-VSAM volume records" (NVRs). If a non-VSAM data set spans volumes, its NVR is in the VVDS of the data set's first volume. Since a BCS is a VSAM data set, it also has a VVR in the VVDS.

Every ICF catalog consists of one BCS and one or more VVDSs. A BCS does not "own" a VVDS: more than one BCS can have entries for a single VVDS. Every VVDS which is connected to a BCS has an entry in the BCS.

For example, Figure 1 shows a possible relationship between two BCSs and three VVDSs on three disk volumes. "BCS.A" has entries for data sets residing on each of the three volumes. "BCS.C" has entries for data sets residing on volumes B and C. Since each volume has data sets cataloged in an integrated catalog facility catalog, each volume contains a VVDS.

BCS.A resides on volume A with VVDS.A. Both the VVDS and the BCS have entries for each other. All three VVDSs are cataloged in BCS.A. BCS.C, residing on volume C, contains entries for VVDS.C and VVDS.B.

Notice that a VVDS has entries for *all* VSAM and SMS-managed data sets on its volume, whereas a BCS can have entries for data sets residing on any volume.
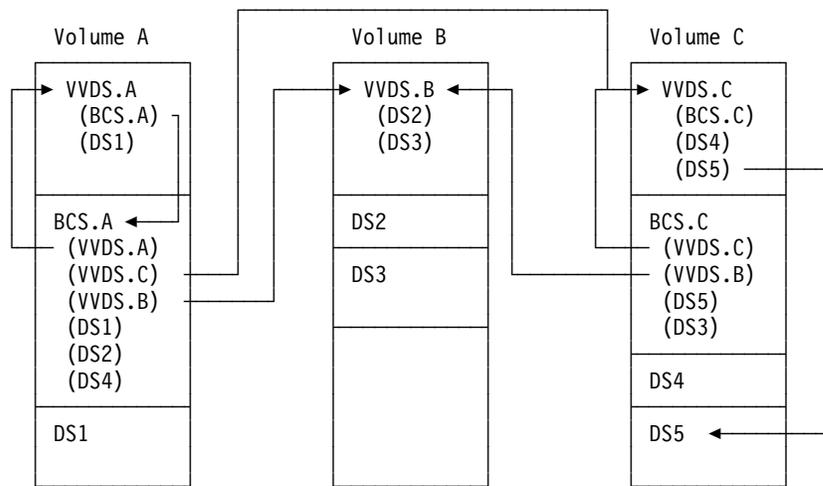


*Figure 1. Relationship of the BCS and the VVDS. Data set names in parentheses are entries in the BCS or VVDS. Only selected arrows are drawn. See the text for a complete explanation of the relationships diagrammed in this figure.*

Figure 1 illustrates how data set entries are contained in both the VVDS and the BCS. Information about a data set is also contained in the VTOC of the volume on which the data set resides, even if the data set is cataloged. To successfully perform all possible operations on a cataloged data set using the catalog, all three elements, the VVDS, BCS, and VTOC, must be synchronized. That is, any equivalent information contained in the BCS and VVDS entries for the data set, and the VTOC DSCB for the data set, must be the same. This is normally done automatically. However, if the catalog components, and the VTOC, become unsynchronized, see "Chapter 7. Analyzing Catalogs for Errors and Synchronization" on page 95 for information on identifying and correcting the problems.

A configuration of ICF catalogs depends on a master catalog. A master catalog has the same structure as any other ICF catalog. What makes it a *master* catalog is that all BCSs are cataloged in it, as well as certain data sets called "system data sets" (for instance, SYS1.LINKLIB and other "SYS1" data sets). Master catalogs are discussed in "The Master Catalog" on page 23.

## Advantages of the Integrated Catalog Facility

The integrated catalog facility offers significant advantages including improved performance, capability, usability, and maintainability.

## Performance

The catalog information that requires the most frequent updates is physically located in the VVDS on the same volume as the data sets, allowing faster access. A catalog request is expedited because fewer I/O operations are needed. Related entries, such as a cluster and its alternate index, are processed together.

## Capability

An ICF catalog can have data sets cataloged on any number of volumes. The BCS can have as many as 123 extents on one volume. One volume can have multiple ICF catalogs on it. All the necessary control information is recorded in the VVDS residing on that volume.

## Usability

With the commands provided, you can reorganize catalogs, move catalogs to devices of different types, merge two catalogs into one, split one catalog into two or more catalogs, share catalogs, and create portable copies.

Significant space savings for generation data groups are achieved in the ICF catalog by reusing space when an old generation is deleted and by using an improved method of recording generation data groups.

## Maintainability

Maintainability is improved by simpler backup and recovery procedures and use of the DIAGNOSE and EXAMINE commands.

### Simpler Backup and Recovery Procedures

The BCS can be maintained independently from the data sets cataloged in it.

The dynamic information associated with a VSAM data set (the data set characteristics) resides in the VVDS on the same volume as the VSAM data set itself. The VVDS contains the data set characteristics that must be synchronized with the data set each time it is updated. Therefore, you can copy the data sets or the volume periodically for backup and recovery without causing the data set and VVDS portion of the ICF catalog to be out of synchronization.

Information that can be out of synchronization in the catalog (for example, a frequently used relative byte address on volume and extents) is moved from the catalog to the VVDS. The VVDS resides on the same volume as the data set component. Therefore, you can back up data sets independently of the BCS.

The BCS maintains a record of the associated data sets with a sphere record. For example, the sphere record contains a record of a base cluster and its related alternate indexes. All the objects associated with the sphere record are processed before the sphere record is updated. You can therefore restart most processing at the point of interruption, without losing data or special processing.

You can also use the Integrated Catalog Forward Recovery Utility (ICFRU) to recover a damaged catalog to a correct and current status. This utility uses system management facilities (SMF) records that record changes to the catalog, updating the catalog with changes made since the BCS was backed up. ICFRU can also be used with shared catalogs or master catalogs, if the master catalog is not being used as a master at the time.

### Error Diagnosis

You can check the structure of a BCS with the EXAMINE command, and the content of a BCS or a VVDS with the DIAGNOSE command.

When analyzing an ICF catalog, you must execute the VERIFY command before the EXAMINE command to verify that the VSAM information for the catalog is current. If the catalog is open in the Catalog Address Space and you execute the EXAMINE command, the data in the catalog may not be current, and incorrect data may be displayed.

With EXAMINE, you can check the structural integrity of a BCS, just as you would check the structural integrity of any other VSAM key-sequenced data set.

With DIAGNOSE, you can compare the BCS and the VVDS and thus verify catalog integrity. If an error is found, DIAGNOSE identifies the problem. Based on the result of the DIAGNOSE output, you can determine how to correct the error.

In addition to the DIAGNOSE command, you can use all the existing access method services and catalog diagnostic aids for the ICF catalog.

Successful execution of a DIAGNOSE command depends on whether the input and compare data sets can be accessed. If you cannot read records from one of these data sets, the DIAGNOSE command terminates with an appropriate message and return code.

For more information on error diagnosis, refer to "Appendix A. Catalog Diagnostic Information" on page 135.

## Using Access Method Services

You use access method services to define and maintain catalogs. Access method services commands can also be used to define and maintain VSAM and non-VSAM data sets. For a complete explanation of the usage of access method services, the required JCL, and examples, see *z/OS DFSMS Access Method Services*.

Table 1 explains the access method services commands that are used with catalogs. For further information about the DEFINE command, see "Using the DEFINE Command" on page 6.

*Table 1. Access Method Services Commands for Catalogs*

| Command | Purpose |
| --- | --- |
| ALTER | Alters previously defined catalog entries or certain catalog attributes, including entries for tape volume catalogs. |
| CREATE | Creates tape library or tape volume entries. |
| DEFINE | Defines catalog entries. |
| DEFINE CLUSTER | Defines VVDSs and catalog entries. |
| DEFINE USERCATALOG | Defines user catalogs including tape volume catalogs (DEFINE USERCATALOG VOLCATALOG). |
| DELETE | Deletes catalog and VVDS entries, data sets, catalogs, VVDSs, and data set control blocks in the VTOC. |
| DIAGNOSE | Determines whether the content of BCS or VVDS records is invalid or unsynchronized. Diagnose recognizes tape library and tape volume record types. It checks the cell structure of the volume catalog. |

*Table 1. Access Method Services Commands for Catalogs  (continued)*

| Command | Purpose |
|---|---|
| EXAMINE | Determines whether structural errors exist in the index or data component of a BCS. |
| EXPORT | Creates a backup copy of a catalog or copy that can be moved to another system or volume. |
| EXPORT DISCONNECT | Disconnects the catalog from the master catalog. |
| IMPORT | Restores an exported backup copy of a catalog, or makes a catalog that was previously exported from one system available for use in another system. |
| IMPORT CONNECT | Connects the catalog to the master catalog on the same system, or on any shared system. |
| LISTCAT | Lists catalog entries including entries for tape volume catalogs. |
| PRINT | Prints data set or catalog records. |
| REPRO | Copies catalogs; splits catalog entries between two catalogs; merges catalog entries into another user catalog or master catalog. REPRO supports tape volume catalogs. |
| VERIFY | Causes a catalog to reflect the end of a data set correctly after an error that prevented closing a VSAM data set. The error might cause the catalog to be incorrect. |

**Restriction:**  The access method services ALTER, CREATE, and DELETE commands should only be used to recover from tape volume catalog errors. Since access method services cannot change the library manager inventory in an automated tape library, ISMF should be used for normal tape library ALTER, CREATE, and DELETE functions.

## JCL Considerations

Under normal conditions, you do not have to specify a catalog on a DD statement when using access method services.

If you are running the Storage Management Subsystem (SMS), you should not use the JOBCAT and STEPCAT DD statements. If a job contains a JOBCAT or STEPCAT DD statement, the first step in the job which refers to an SMS-managed data set fails, and all following steps in that job are bypassed. The job terminates.

You might have used these DD statements to recatalog an incorrectly cataloged data set, or to define new system data sets. However, these tasks can normally be accomplished by specifying a catalog in the CATALOG parameter of an access method services command or by using the SYS% conversion facility.

If the aliases of your catalogs are carefully defined, your users should never need to use a JOBCAT or STEPCAT DD statement to specify a catalog.

The examples in this book occasionally use SMS functions. However, most examples can be used in either an SMS or a non-SMS environment, although minor changes might be necessary. Where allocation information can be included in a data class (for example, space attributes), they are explicitly coded. If management class or storage class is specified, it can be removed for non-SMS environments.

**Note:**  If an ICF catalog resides on a volume with a UCB address above the 16MB line, JOBCAT and STEPCAT are not supported.

See *z/OS DFSMS: Using Data Sets* for more details on allocating SMS-managed data sets. See *z/OS DFSMS Access Method Services* for more details on JCL requirements for access method services, and for information on using JOBCAT and STEPCAT.

# Using the DEFINE Command

Data sets of all types can be cataloged in an ICF catalog. This can be accomplished through JCL, dynamic allocation, access method services, IEHPROGM, or other utilities. If you are using the Storage Management Subsystem, data sets are automatically cataloged.

When you issue the access method services DEFINE command to create a VSAM object, a catalog entry is built that describes the object. Before an object can be defined, there must be a catalog in which to define the object. The catalog is chosen according to the catalog search order.

When you define a VSAM data set or object, you specify attributes to be associated with it. The attributes include, for example, the SMS Management Class name to be associated with the data set.  After the object is defined, it can be processed with other access method services commands and with application programs.

Password protection is ignored for the protection of any data set cataloged in an ICF catalog, including the catalogs themselves. This is a change from previous releases where passwords were ignored only for SMS-managed data sets.

When you define a catalog, cluster, or alternate index, you can specify attributes in several different ways. The parameter set for DEFINE USERCATALOG, MASTERCATALOG, CLUSTER, and ALTERNATEINDEX is directly related to the way the attributes are stored in the ICF catalog. The catalog entries that describe an ICF catalog, cluster, or alternate index are:

- The cluster entry, which describes the attributes of the cluster or catalog.
- The alternate index entry, which describes the attributes of the alternate index.
- The data entry, which describes the attributes of the data component of a catalog, cluster, or alternate index.
- The index entry, which describes the attributes of the index component of a catalog, key-sequenced cluster, or alternate index.

When you specify attributes as parameters of USERCATALOG, MASTERCATALOG, CLUSTER, or ALTERNATEINDEX, consider the following:

- Under SMS, you can specify a data class, management class, or storage class. You can specify data set size in kilobytes or megabytes as well as cylinders, tracks, or records.
- Attributes specified in the parameters are defined in the cluster or alternate index entry of an ICF catalog if they pertain to that entry.
- Attributes specified in the parameters are defined for the data or index entries to which they pertain.
- If the same attribute is specified as a subparameter of DATA or INDEX, the value of the attribute specified at the DATA or INDEX level overrides the value of the attribute specified at the USERCATALOG, MASTERCATALOG, CLUSTER, or ALTERNATEINDEX level.

You can use the LISTCAT command with the ALL option to list catalog entries and to determine what various attributes are stored in the catalog. See *z/OS DFSMS Access Method Services* for a description of the attributes for each type of entry.

## Notational Conventions

A uniform notation is used to describe the syntax of commands, or the format of control records. This notation is not part of the language. The following conventions can be used in this publication:

**[ ]**　　Brackets enclose an optional entry. You can, but need not, include the entry. Examples are:
- **[*length*]**
- **[MF=E]**

**|**　　An OR sign (a vertical bar) separates alternative entries. You must specify one, and only one, of the entries unless you allow an indicated default. Examples are:
- **[REREAD|LEAVE]**
- **[*length*|'S']**

**{ }**　　Braces enclose alternative entries. You must use one, and only one, of the entries. Examples are:
- **BFTEK={S|A}**
- **{K|D}**
- **{*address*|S|O}**

Sometimes alternative entries are shown in a vertical stack of braces. An example is:

**MACRF={(R[C|P])}**
**{(W[C|P|L])}**
**{(R[C],W[C])}**

In the example above, you must choose only one entry from the vertical stack.

**. . .**　　An ellipsis indicates that the entry immediately preceding the ellipsis can be repeated. For example:
- **(*dcbaddr*,[(*options*)],. . .)**

**' '**　　A ' ' indicates that a blank (an empty space) must be present before the next parameter.

**REQUIRED KEYWORDS AND SYMBOLS**
Entries shown **IN THE FORMAT SHOWN HERE** (notice the type of highlighting just used) must be coded exactly as shown. These entries consist of keywords and the following punctuation symbols: commas, parentheses, and equal signs. Examples are:
- **CLOSE , , , ,TYPE=T**
- **MACRF=(PL,PTC)**

The format (the type of highlighting) that is used to identify this type of entry depends on the display device used to view a softcopy book. The published hardcopy version of this book displays this type of entry in uppercase boldface type.

**DEFAULT VALUES**

Values shown **IN THE FORMAT SHOWN HERE** (notice the type of highlighting just used) indicate the default used if you do not specify any of the alternatives. Examples are:

- **[EROPT={ACC|SKP|ABE}]**
- **[BFALN={F|D}]**

The format (the type of highlighting) used to identify this type of entry depends on the display device used to view a softcopy book. The published hardcopy version of this book displays this type of value in underscored uppercase boldface type.

*User Specified Value*

Values shown *in the format shown here* (notice the type of highlighting just used) indicate a value to be supplied by you, the user, usually according to specifications and limits described for each parameter. Examples are:

- *number*
- *image-id*
- *count*

The format (the type of highlighting) that is used to identify this type of entry depends on the display device used to view a softcopy book. The published hardcopy version of this book displays this type of value in lowercase italic type.

# Chapter 2. Planning a Configuration of Catalogs

Before you define a catalog, there are some major issues which you should consider carefully. These issues involve not only the specific characteristics of the catalog, but also the characteristics of the configuration of catalogs that will reside on your system.

This chapter only addresses issues related to the complete catalog configuration. Catalog protection, size, and performance attributes are discussed in other chapters.

## Using Catalogs with the Storage Management Subsystem

The Storage Management Subsystem (SMS) is a subsystem of DFSMS which automates data management, including data placement on volumes, cataloging, and backing up data.

## Catalog Rules for Data Managed by the Storage Management Subsystem

Under SMS, all permanent data sets must be cataloged. Only integrated catalog facility type catalogs can be used with SMS. Although a catalog contains entries for SMS-managed data, the catalog itself does not have to be SMS-managed, but it is recommended that catalogs containing entries for SMS-managed data also be SMS-managed. A catalog can contain entries for data sets which are SMS-managed, and entries for other data sets which are not managed by SMS.

All new SMS-managed data sets are cataloged when they are allocated, not when the job step terminates. SMS-managed generation data sets are cataloged in a "deferred roll-in" status at allocation, and then are "rolled-in" to the generation data group at step termination, if their disposition indicates they are to be cataloged. To set the disposition, use DISP=(NEW,CATLG).

You cannot use password protection for ICF catalogs or data sets cataloged in an ICF catalog. Although you can define catalogs and data sets with passwords, SMS ignores the passwords. Use RACF™, a component of the SecureWay Security Server for z/OS, to protect data that is cataloged in an ICF catalog.

There are special RACF facility classes which are used with SMS. You need to define these classes to restrict the use of certain functions. For instance, you should not normally be able to specify a catalog to be searched unless you have authorization for the RACF directed catalog facility class. See "Storage Administration (STGADMIN) Profiles in the FACILITY Class" on page 76 for more information.

All SMS-managed data sets have entries in the VVDS. Non-VSAM data sets have non-VSAM volume records (NVRs); VSAM data sets have VSAM volume records (VVRs). This requires larger VVDSs on SMS-managed volumes.

Temporary VSAM data sets which are SMS-managed also have VVDS entries, although they do not have BCS entries.

For more information on the Storage Management Subsystem, see *z/OS DFSMSdfp Storage Administration Reference* . For more information on catalog considerations when moving to an SMS environment, see *z/OS DFSMS: Implementing System-Managed Storage*.

## Defining Catalog and VVDS Data Classes

You can simplify the definition of new BCSs and explicitly defined VVDSs by creating data classes for each. The data classes should specify the characteristics most commonly used at your installation. You can override most attributes defined in the data class when you issue the DEFINE command. You can use a data class to define the catalog even if the catalog is not SMS-managed.

For BCSs, choose appropriate default size and performance attributes. See "Estimating Space Requirements for the BCS" on page 43, "Choosing Options to Adjust Catalog Performance" on page 46, and "Defining a Basic Catalog Structure" on page 47 for recommendations for BCS attributes. For RECORG, specify KS. Do not specify a key length or offset, or a logical record length. These values are set by the DEFINE USERCATALOG command.

For VVDSs, choose an appropriate size, usually 10 tracks primary and secondary space. There are no special performance attributes for VVDSs. For RECORG, specify ES. Allow DEFINE CLUSTER to set all other values.

## Defining a Catalog Management Class

You can define a special management class for BCSs. This management class can be shared with other critical system data sets which have the same requirements, to ensure that recent backup copies are available in case a catalog must be recovered.

In general, the management class you specify for BCSs should have the following attributes:
1. For expiration, NOLIMIT for all attributes.
2. For migration, NONE for command or auto migration.
3. For backup,
   a. For backup frequency, specify 0 to have DFSMShsm™ back up the catalog every occasion that it is changed.
   b. For administrator or user command backup, specify ADMIN to limit backup operations to storage administrators.
   c. For auto backup, specify YES.

Choose appropriate values for other backup attributes, depending on the needs of your installation. It is generally prudent to keep extra backup copies of catalogs, even after the catalog has been deleted.

Management class does not apply to VVDSs. VVDSs should only be backed up as part of a full volume dump.

## Defining a Catalog Storage Class

Storage class is used to define the performance objectives for a data set. Since catalogs have their own cache, specification of additional performance objectives through storage class is unnecessary.

If you want to control the placement of your catalogs, assign them a storage class with the Guaranteed Space attribute. The Availability objective for catalogs should be standard, although you can make it Continuous for the catalog of a critical application.

# Sharing Catalogs Among Systems

A shared catalog is a basic catalog structure which is eligible to be used by more than one system. It must be defined with SHAREOPTIONS(3 4), and reside on a shared volume. A DASD volume is initialized as shared using MVSCP or the MVS hardware configuration definition (HCD) facility. Note that the device must be defined as shared to all systems that access it. If some systems have the device defined as shared and some do not, catalog corruption will occur. Check with your system programmer to determine shared volumes. Note that it is not necessary that the catalog actually be shared between systems; the catalog address space assumes it is shared if it meets the criteria stated above. All VVDSs are defined as shared. Tape volume catalogs can be shared in the same way as ICF catalogs.

By default, catalogs are defined with SHAROPTIONS(3 4). You can specify that a catalog is not to be shared by defining the catalog with SHAREOPTIONS(3 3). Only define a catalog as unshared if you are certain it will not be shared. Place unshared catalogs on volumes which have been initialized as unshared. Catalogs which are defined as unshared and which reside on shared volumes will become damaged if referred to by another system.

If a catalog is shared, its catalog address control structures are refreshed when updates are made to the catalog from any system. This ensures that each system is using an up-to-date copy of the catalog at all times.

# Catalog Sharing Protocols

There are two protocols that are used to share catalogs between systems:
* VVDS mode
* Enhanced Catalog Sharing (ECS) mode

### VVDS Mode Sharing
There is information necessary to communicate changes in a basic catalog structure (BCS) to other systems that are sharing the catalog. This information is stored in a special record in the VVDS on the volume the catalog is defined on. The information is used to ensure the consistency of the catalog records that are cached on any sharing subsystem. It is also used to update the BCS control block structure in those cases where a sharing system has extended beyond the current high used value or to a new extent. In addition, it is used to invalidate BCS data and index buffers when they have been updated from a sharing system.

The storing and retrieval of this information requires additional I/O to the volume containing the catalog. In some cases, this I/O overhead can become significant and have a noticeable impact on system or sysplex performance. VVDS Mode sharing is the default mode of sharing, and is the only protocol that is supported by systems prior to DFSMS V1 R5.

### Enhanced Catalog Sharing (ECS) Mode Sharing
This sharing protocol stores the information that describes changes to a shared catalog in the Coupling Facility. The I/O required for the VVDS mode protocol is eliminated, resulting in better sysplex-wide performance.

This protocol is used when three conditions are satisfied:

- The ECS cache structure is defined in a coupling facility that uses an installation-defined Coupling Facility Resource Manager (CFRM) policy
- A successful connection has been made to the ECS cache structure
- A catalog is referenced that has the ECSHARING attribute set

For additional information on enabling this protocol, see "Using Enhanced Catalog Sharing Mode" on page 36.

## Sharing Catalogs Under the Storage Management Subsystem

A catalog can contain entries for SMS-managed and unmanaged data sets. A catalog containing entries for unmanaged data can itself be managed.

A catalog which contains entries for SMS-managed data does not have to reside on SMS-managed volumes. Likewise, a catalog which contains entries for data not managed by SMS can itself reside on an SMS-managed volume.

## Preventing Lockouts on Shared Volumes

CATALOG MANAGEMENT uses the SYSIGGV2 reserve while serializing access to catalogs. The SYSIGGV2 reserve is used to serialize the entire catalog BCS component across all I/O as well as to serialize access to specific catalog entries. The SYSZVVDS reserve is used to serialize access to associated VVDS records. The SYSZVVDS reserve along with the SYSIGGV2 reserve provide an essential mechanism to facilitate cross system sharing of catalogs.

When your data sets reside on the same volume as other shared catalogs, deadlocks can occur. An example is shown in Figure 2 on page 13. In this example, SYS 1 and SYS 2 share DASD volumes, VOLSER1 and VOLSER2. The SYSIGGV2 reserve is held for CATALOG A by SYS 1 while trying to obtain a reserve for data set A. SYS 2 has a SYSIGGV2 reserve for CATALOG B while trying to obtain a reserve for data set B. Reserves for data set A or data set B could be for SYSVTOC or SYSZVVDS. You can prevent such deadlocks, by *always* converting the SYSIGGV2 reserve to a SYSTEMS ENQUEUE using Global Resource Serialization (GRS) or an equivalent product. It is not necessary to convert the SYSZVVDS reserve. The SYSZVVDS reserve is issued with a scope of SYSTEMS. For this reason, it must be excluded from GRS processing. For further information concerning catalog serialization, see *z/OS MVS Planning: Global Resource Serialization*.
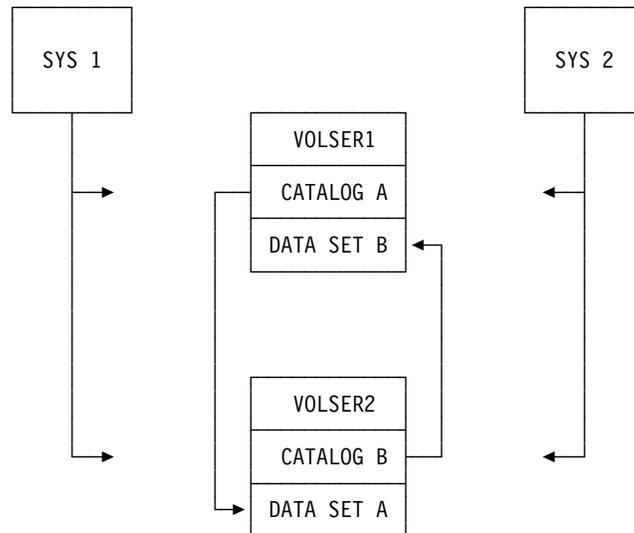
*Figure 2. Data Sets Residing on the Same Volume as Other Shared Catalogs*

Lockouts that occur because of a failure to convert the SYSIGGV2 reserve are not system failures for which the IBM® System Support personnel are able to provide assistance. You prevent these lockouts by ensuring that your support personnel has converted the SYSIGGV2 reserve.

## Sharing Catalogs Among Systems With Different Levels of DFSMS

You can share catalogs among systems which are running different levels of DFSMS. "Sharing Catalogs Under the Storage Management Subsystem" on page 12. For specific information on sharing catalogs between different levels of DFSMS, see *z/OS DFSMS Migration*.

## Using the SYS% Conversion Facility

The purpose of the SYS% facility is to provide a method of managing system data sets with SMS in a multi-host environment. Each system requires duplicate system data sets (data sets with the high-level qualifier SYS1).

Because you cannot use JOBCAT and STEPCAT DD statements with SMS, and because SMS requires uniquely named data sets, you normally would not be able to access the system data sets for one system from another system. However, you might need to update one system's SYS1 data sets from a different system.

The SYS% facility allows you to do this by permitting access to system data sets with alias names. The alias names have SYS for the first three characters, followed by another character which is *not* 1 (for example, SYSA). You use the alias to first orient the search to the appropriate master catalog, which is then searched for the requested system data set.

As an example, consider a complex with two systems, each using SMS, called SYSTEMA and SYSTEMB. To connect the systems, the master catalog of each system is defined as a user catalog in the master catalog of the other system. This can be seen in Figure 3 on page 14.

```
        SYSTEMA                              SYSTEMB

  ┌─────────────────────┐            ┌─────────────────────┐
  │    MASTCAT.A         │      ┌──►  │    MASTCAT.B         │
  │  ┌───────────────┐   │      │     │  ┌───────────────┐   │
  │► │ (MASTCAT.B)   │───┼──────┘     │  │ (MASTCAT.A)   │   │
  │┌─┤ (SYSB) alias  │   │            │  │ (SYSA) alias  │   │
  ││ │ (SYS1.LINKLIB)│   │            │┌─┤ (SYS1.LINKLIB)│   │
  ││ └───────────────┘   │            ││ └───────────────┘   │
  ││                     │            ││                     │
  ││ ┌──►               │            ││ ┌──►               │
  │└─┤    SYS1.LINKLIB   │            │└─┤    SYS1.LINKLIB   │
  │                      │            │                      │
  └─────────────────────┘            └─────────────────────┘
```
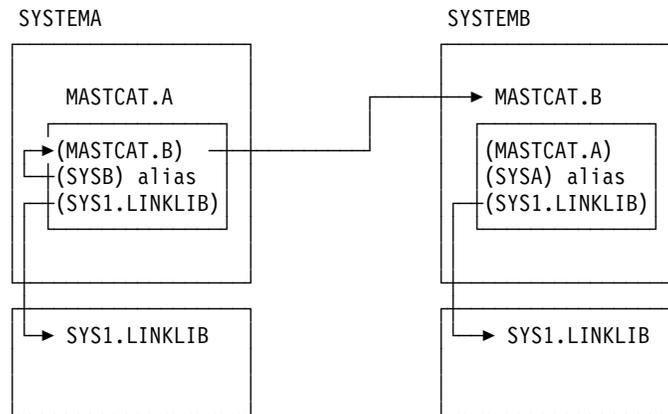
*Figure 3. Master Catalogs of Connected SMS Systems. The pointers from MASTCAT.B to MASTCAT.A are not shown, to make the diagram clearer. Catalog entries are enclosed in parentheses.*

In this example, MASTCAT.A is defined with the alias SYSA in MASTCAT.B; and MASTCAT.B is defined with the alias SYSB in MASTCAT.A. Each catalog contains a pointer to a different SYS1.LINKLIB system data set. If a job addresses the data set SYS1.LINKLIB, the SYS1.LINKLIB which is allocated depends on which system is running the job.

In order for a job running on SYSTEMA to access the SYS1.LINKLIB cataloged in MASTCAT.B on SYSTEMB, that job must orient itself to the other master catalog, without using JOBCAT or STEPCAT. This can be done by using the SYS% facility and the alias defined for the other master catalog.

If the SYS% facility is active, a job running on SYSTEMA can address SYSB.LINKLIB. Because SYSB is an alias of MASTCAT.B, MASTCAT.B is searched for a data set named SYSB.LINKLIB. Because one is not found, MASTCAT.B is again searched, only this time for SYS1.LINKLIB, which is found. The job then uses the SYS1.LINKLIB cataloged in MASTCAT.B, even though the job ran on SYSTEMA, and there is a SYS1.LINKLIB cataloged in SYSTEMA's master catalog.

If the alias is not defined (if you specify an alias that does not exist), the SYS% conversion will resolve to the current system. For example, if you specify SYSX.LINKLIB on SYSTEMA, and SYSX is not defined as an alias to another system, then the request would go to SYS1.LINKLIB on SYSTEMA . Make sure that you have located the correct system SYS1 data set.

## Allocating New System Data Sets Under SMS

The SYS% facility is specifically used for allocating existing data sets. It is not meant to be used to allocate new data sets, for example, for a new system. However, the presence of SYS% aliases used with the SYS% facility allows for the allocation of new system data sets without using a JOBCAT or STEPCAT DD statement.

You can also avoid using JOBCAT and STEPCAT when the UCB is above the 16MB line by using DEFINE PAGESPACE RECATALOG with the CATALOG parameter. With this method, you can recatalog the PAGE data set into the desired catalog by completing the following steps:

1. Create new page volumes using the DEFINE PAGESPACE command.

2. Move the new pagespace entries from the driver master catalog to the new master catalog using the REPRO MERGCAT command as shown in the following example:

```
//SYSPRINT DD  SYSOUT=*
//PLPA     DD  UNIT=3390,VOL-SER=S0HP22,
//            DISP=OLD
//SYSIN    DD *
   DEFINE  PAGESPACE  -
            ( FILE(PLPA) -
              NAME (SYS1.newsysname.PLPA) -
              CYLINDERS (1) -
              UNIQUE -
              VOLUME (S0HP22)  )
              CYLINDERS(1)
              UNIQUE -
              VOLUME(S0HP22)
//MOVEPAGE EXEC  PGM=IDCAMS
//UCATDVR  DD DSN=CATALOG.drivercat,DISP=SHR
//UCATNEW  DD DSN=CATALOG.newcat,DISP=SHR
//DD1      DD VOL=SER=S0HP22,UNIT=3390,DISP=OLD
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
   REPRO ENTRIES(SYS1.newsysname.*) -
```

If the new system is SYSTEMB, and MASTCAT.B has been defined with the alias SYSB in MASTCAT.A, you can run allocation jobs on SYSTEMA to allocate data sets with the high-level qualifier SYSB. Once all the new data sets have been allocated (or at any time that you are ready), you can do a generic rename of the data sets, changing the SYSB qualifier to SYS1:

```
ALTER SYSB.*  NEWNAME(SYS1.*)  CATALOG(MASTCAT.B)
```

To successfully alter the names of the SYSB data sets from SYSTEMA, you must have appropriate RACF authority to the directed catalog facility class. See "Storage Administration (STGADMIN) Profiles in the FACILITY Class" on page 76.

This process of allocating new data sets with the high-level qualifier SYSB, and then renaming them by directing the catalog request, does not use the SYS% facility. After renaming the system data sets, MASTCAT.B does not contain any SYSB data sets. However, the SYS1 data sets in MASTCAT.B can be accessed by allocating them using the SYSB high-level qualifier, if the SYS% facility is activated.

### Enabling SYS% Conversion
The setting for the SYS% facility ("on" or "off") is set in the SYSCAT*xx* member of SYS1.NUCLEUS. You can change this setting without an IPL by using the MODIFY CATALOG operator command:

```
MODIFY CATALOG,SYS%ON  or  MODIFY CATALOG,SYS%OFF
```

Using the MODIFY command, you can use the SYS% facility selectively, only turning it on when you need it. See "Identifying the Master Catalog and Initial Configuration (SYSCATxx)" on page 31 for information on activating the SYS% facility at IPL.

### Restrictions on Using SYS% Conversion
The SYS% facility should only be used to allocate existing data sets. The SYS% facility can be used to build systems in an environment where the new and old data sets with the same names need to be cataloged in different catalogs at allocation time. For this reason, several functions are not supported in SYS% processing. Attempting them can cause undesired results or errors.

Do not attempt to use the SYS% facility to process SYS1 data sets with the DEFINE or ALTER commands. However, there is no restriction on using DEFINE or ALTER on data sets whose *real* data set name has a high level qualifier which can later be used by SYS%. For example, you can define a data set named SYSB.LINKLIB, and if SYSB is an alias to another system's master catalog, it is defined in that master catalog *under the name SYSB.LINKLIB*. Renaming the data set can then be accomplished if you have appropriate RACF authority to the directed catalog FACILITY class profile.

Errors result if you expect DEFINE, ALTER, or LISTCAT LEVEL to convert a SYS% alias to SYS1.

You can use the SYS% facility to delete data sets only if you use their fully-qualified data set names (a "discrete" deletion). You can do this during disposition processing or through access method services. For example, if the real data set name is "SYS1.LINKLIB", you can allocate and delete the data set using "SYSB.LINKLIB" if the alias "SYSB" is properly defined. Do not attempt to generically delete a group of SYS1 data sets using a SYS% alias by specifying "SYSB.*".

Do not use indirect volume serial numbers (for example, ******) when using SYS% conversion. Use specific volume serial numbers for all references to SYS1 data sets. Indirect volume serial numbers resolve to the IPL volume of the system running the job.

Be careful if you attempt to use SYS% conversion with the ISPF/PDF utilities, ISMF, and TSO commands. Unexpected results can occur, since these programs sometimes use generic functions.

## Integrity of Shared Catalogs and VVDSs

The catalog address space maintains information about when a catalog, either a BCS or a VVDS, is changed. Each system can identify when a catalog has changed, and a system refreshes its copy of the catalog in main memory only when necessary.

Sharing systems maintain control blocks for each BCS and VVDS accessed. Under normal conditions, these control blocks need no special maintenance.

However, if you cannot access a shared BCS or VVDS after recovery, refresh the control blocks for the BCS or VVDS. Use the MODIFY CATALOG command with CLOSE or UNALLOCATE to remove the control blocks for a BCS. The next request that accesses the BCS rebuilds the control blocks. The control blocks for a VVDS can be removed using the VCLOSE or VUNALLOCATE options of MODIFY CATALOG.

If you recover a shared BCS to a volume with a different volume serial number or device type than the original BCS had, use the access method services IMPORT CONNECT ALIAS command to update the catalog connector record for the BCS in the master catalog of each sharing system.

Sharing catalogs also affects catalog performance. See "The Effect of Sharing Catalogs on Cache Usage" on page 30 for more information on the performance of shared catalogs, and "Chapter 8. Working with the Catalog Address Space" on page 103 for more information on MODIFY CATALOG.

# Extended Alias Support

Alias entries are defined to allow a reference to a particular name to be translated to a different name for the actual data. This provides a means for users to access their data by a particular name without having to know the actual name of the data set that contains their data. This technique is particularly useful for migration from release to release of products.

For example, the names of libraries for different releases or versions of products can contain the release or version information. An alias can be created without this information, so users are not aware when the underlying library changes. This minimizes the impact on job control language and TSO changes as a result of migrations.

As an example, if SYS1.V1R3M0.PRODUCT is the name of a product library, an alias of SYS1.PRODUCT is created by this IDCAMS DEFINE command:

```
DEFINE ALIAS (NAME(SYS1.PRODUCT) RELATE(SYS1.V1R3M0.PRODUCT))
```

Users can now reference SYS1.PRODUCT, and if version 1 release 4 of the product is installed, the alias name can simply be changed to refer to the new library. This allows the new version of the library to be tested without disrupting current users.

A particular problem occurs in environments that share catalogs, particularly master catalogs. Different systems might be at different levels of software. Users would like to use the alias approach to minimize the effect of data set name changes. However, the actual data set name had to be specified when the ALIAS was defined; therefore all systems would see the same value.

A parameter for the DEFINE ALIAS command, SYMBOLICRELATE, allows the specification of the base data set name using system symbols. The above example could then be defined as:

```
   DEFINE ALIAS (NAME(SYS1.PRODUCT) -
       SYMBOLICRELATE('SYS1.&PRODVR..PRODUCT'))
```

If the system symbol '&PRODVR' was set  to 'V1R3M0' on System A, and 'V1R4M0' on System B, then a reference to the name SYS1.PRODUCT would access the proper data set, depending on what system the alias name was used from. To set system symbols, see the *z/OS MVS Initialization and Tuning Reference*. In this case, the alias name is resolved at the time of use, rather than at the time of definition. As sharing systems are ready to upgrade to the new data set, they only need change the definition of the appropriate system symbol or symbols to access the new data set by the original alias.

This support is available for defining aliases of user catalogs, or for non-VSAM (non-GDS) data sets. If the string containing the system symbols cannot be resolved (for example, the symbol might not be defined on the referencing system), the reference will probably fail. This is because there would be no data set name that matched the value specified in the SYMBOLICRELATE keyword.

The actual resolution of the symbolic string to a data set name is done at two possible times:

- If the resolved name is an ICF catalog, it is done at catalog address space initialization or when the multi-level alias table is reinitialized
- If the resolved name is a non-VSAM data set, it is done at the time of reference to the data set by a catalog request

# The Catalog Search Order

When an application or user refers to a cataloged data set or creates a data set which is to be cataloged, the configuration of catalogs must be searched to find the appropriate catalog. The catalog is chosen according to a search order determined by the aliases defined for the catalogs, the names of the catalogs, and the multilevel alias search level.

# The Standard Search Order

Most catalog searches should be based on catalog aliases. When appropriate aliases are defined for catalogs, the high-level qualifier of a data set name is identical to a catalog alias and identifies the appropriate catalog to be used to satisfy the request. However, some alternatives to catalog aliases are available for directing a catalog request, specifically the JOBCAT and STEPCAT DD statements, the CATALOG parameter of access method services, and the name of the catalog.

Try to avoid using JOBCAT or STEPCAT, as they have functional restrictions that can cause problems. They are not supported for any job step that contains a reference to SMS-managed data sets. For example, catalogs that are located on devices whose unit control blocks (UCBs) are defined above the MB line cannot be referenced by a JOBCAT or STEPCAT. Instead, use aliases (particularly multi-level aliases) or the CATALOG parameter to handle the functions that are provided by JOBCAT and STEPCAT.

Table 2 on page 18 summarizes the catalog search order for defining or locating a data set.

*Table 2. Standard Search Order for Catalog Requests*

| Defining a Data Set | Locating a Data Set |
|---|---|
| 1. Use the catalog named in the IDCAMS CATALOG parameter, if coded. | 1. Use the catalog named in IDCAMS CATALOG parameter, if coded. If the data set is not found, fail the job. |
| 2. If the data set is a generation data set, the catalog containing the GDG base definition is used for the new GDS entry. | 2. If the data set is a generation data set, the catalog containing the GDG base definition is used for the new GDS entry. |
| 3. Otherwise, use the catalog named in the STEPCAT DD statement. | 3. Otherwise, search all catalogs specified in a STEPCAT DD statement in order. |
| 4. If no STEPCAT, use the catalog named in the JOBCAT DD statement. | 4. If not found, search all catalogs specified in a JOBCAT DD statement in order. |
| 5. If no JOBCAT, and the high-level qualifier is a catalog alias, use the catalog identified by the alias or the catalog whose name is the same as the high-level qualifier of the data set. | 5. If not found, and the high-level qualifier is an alias for a catalog, search the catalog or if the high-level qualifier is the name of a catalog, search the catalog. If the data set is not found, fail the job. |
| 6. If no catalog has been identified yet, use the master catalog. | 6. Otherwise, search the master catalog. |

**Restrictions:**

1. For SMS-managed data sets, JOBCAT and STEPCAT DD statements are not allowed, and cause job failure. You can use RACF to prevent the use of the CATALOG parameter and restrict the ability to define data sets in the master catalog.

2. For DEFINE USERCATALOG, the catalog will be added to the master catalog of the running system whether the CATALOG parameter is used or not, or if JOBCATS or STEPCATS are provided or not.

To use an alias to identify the catalog to be searched, the data set or object name, or the generation data group base name, must be a qualified name.

If you are using a JOBCAT DD statement to specify a job catalog, or a STEPCAT DD statement to specify a step catalog, the search order is modified. If a step has a catalog assigned (with a STEPCAT DD statement), the step catalog is searched first. If you are cataloging a data set, the data set is cataloged in the step catalog. If you are referring to an existing data set, and the data set is not in the step catalog, the job catalog is searched.The JOBCAT DD statements and the STEPCAT DD statements are ignored for the DEFINE USERCATALOG command with the CATALOG parameter, regardless of the use of STGADMIN.IGG.DIRCAT.

Job catalogs are used in the same manner as step catalogs. However, the JOBCAT and STEPCAT DD statements cannot be used in an SMS environment. If catalog aliases are appropriately defined, these DD statements are not necessary.

The catalog search order is also modified when the CATALOG parameter of access method services commands is used to direct the catalog request. When you specify a catalog in the CATALOG parameter, and you have appropriate RACF authority to the FACILITY class profile STGADMIN.IGG.DIRCAT, the catalog you specify is used.

For instance, DEFINE USERCATALOG CATALOG(SYS1.MASTER.ICFCAT) defines a catalog with a connector record in SYS1.MASTER.ICFCAT, even if SYS1.MASTER.ICFCAT is not the master catalog on the system where you issue the command. See "Storage Administration (STGADMIN) Profiles in the FACILITY Class" on page 76 for more information on the RACF directed catalog profile.

There can be a significant performance impact when using JOBCAT and STEPCAT DD statements. Multiple catalogs might be searched to locate entries, compared to a single catalog for a data set reference accessible from the master catalog.

# The Multilevel Alias Facility

You can augment the standard catalog search order by defining multilevel catalog aliases. A multilevel catalog alias is an alias of two or more high-level qualifiers. You can define aliases of up to four high-level qualifiers.

However, the multilevel alias facility should only be used when a better solution cannot be found. The need for the multilevel alias facility can indicate poor data set naming conventions. Before defining multilevel aliases, review your data set naming conventions. See *MVS/ESA SML: Managing Data* for more information on data set naming conventions.

## Overview of the Multilevel Alias Facility

Under the standard catalog search order, when you are using aliases, you might find that certain catalogs become over used. For instance, if the catalog becomes too large, recovery might become excessively disruptive, or the catalog might be required to process an excessive number of requests, resulting in frequent enqueues. If these catalogs contain large numbers of data sets with the same high-level qualifier, you can alleviate the problem with the multilevel alias facility. Using multilevel aliases, you can have data sets with the same high-level qualifier cataloged in *different* catalogs, without using JOBCAT or STEPCAT DD statements.

For example, your installation might have a project, PROJECT1, which requires many data sets and applications. Perhaps there are large collections of PROJECT1

data sets which have the same second qualifier: TEST or PROD. There are also some miscellaneous PROJECT1 data sets with neither TEST nor PROD as the second qualifier.

In this case, you might define catalog aliases as follows:
- Alias PROJECT1.TEST for catalog SYS1.ICFCAT.PRO1TEST,
- Alias PROJECT1.PROD for catalog SYS1.ICFCAT.PRO1PROD, and
- Alias PROJECT1 for catalog SYS1.ICFCAT.PROJECT1.

If the alias search level is 2, then data sets are cataloged as follows:

| Data Set | Catalog | Reason |
|---|---|---|
| PROJECT1.UTIL.CNTRL | SYS1.ICFCAT.PROJECT1 | The second qualifier is neither TEST nor PROD. |
| PROJECT1.PROD.DATA | SYS1.ICFCAT.PRO1PROD | There are two qualifiers, and the two qualifiers form the alias PROJECT1.PROD. |
| PROJECT1.PROD | SYS1.ICFCAT.PROJECT1 | There is only *one* qualifier in this data set name: PROJECT1. PROD is not a qualifier, so it is not used in the multilevel alias search. |
| PROJECT1.TEST.CNTRL | SYS1.ICFCAT.PRO1TEST | There are two qualifiers, and the two qualifiers form the alias PROJECT1.TEST. |
| PROJECT1.TEST.A.B | SYS1.ICFCAT.PRO1TEST | The first two qualifiers are used as the alias, since the search level is 2. The third qualifier is not used in the search. |

In this example, programs being tested (TEST) are isolated from production programs and data (PROD) and other miscellaneous files. This isolation is desirable for data protection and availability. Backup and recovery of one catalog would not affect projects using the other catalogs.

The alias search level is specified in the SYSCAT*xx* member of SYS1.NUCLEUS or LOAD*xx* member of SYS1.PARMLIB (see "Identifying the Master Catalog and Initial Configuration (SYSCATxx)" on page 31). It can also be changed without an IPL using the MODIFY CATALOG,ALIASLEVEL operator command (see "Changing the Multilevel Alias Search Level" on page 122).

When you use an alias to select catalogs, the use of a single level catalog name can also affect the selection. When a catalog has a single level name, that name behaves as an alias. For example, if a catalog has the name of ICFCAT, data sets beginning with ICFCAT such as ICFCAT.DATA1, ICFCAT.DATA2, and ICFCAT.VSAM.DATA are found in the ICFCAT catalog if JOBCAT, STEPCAT or the CATALOG parameter is not used to limit the search. See "Using LISTCAT in Examples" on page 55 for an example using the CATALOG parameter.

With the multilevel alias facility, this concept is extended to catalogs whose names have multiple levels. For instance, if MULTILEVELALIAS=2, and a catalog ICF.CAT exists, ICF.CAT will also behave as an alias. Data sets such as ICF.CAT.DATA1, ICF.CAT.DATA2, and ICF.CAT.VSAM.DATA are found in the ICF.CAT catalog if no JOBCAT, STEPCAT, or CATALOG parameter is used.

Using the multilevel alias facility and having short catalog names can influence generic searches for catalogs. The list of catalogs to be searched is more complex when a generic entry is not fully qualified. The generic portion of the name is searched first before all alias levels are satisfied. For example, if the name is SYS1.* and MULTILEVELALIAS=2, catalog management does not know which specific catalogs to select and selects all aliases and catalogs beginning with SYS1 and having two levels. For example, SYS1.CAT and SYS1.ALIAS could be selected. This list is then used to select the catalogs to be searched. Duplicates on the list are eliminated and then each catalog is searched for entries that match the search parameter. If SYS1.ALIAS points to ICF.CAT, then catalogs SYS1.CAT and ICF.CAT will be searched. SYS1.CAT is selected because its name matches the generic SYS1.*. ICF.CAT is selected because an alias that points to it, has a name that matches the generic SYS1.*.

Duplicate entries can appear if this alternate selection occurs. If both SYS1.CAT and ICF.CAT contain an entry for SYS1.PROCLIB., SYS1.PROCLIB will be returned twice. SYS1.PROCLIB is returned once for SYS1.CAT where it was found, and once for ICF.CAT where it was also found. You should determine which catalogs are involved because it might appear that you are receiving duplicate entries, but you can be receiving different entries with the same name from different catalogs.

## Choosing Aliases and an Alias Search Level

You should choose multilevel aliases carefully to avoid problems when trying to locate the data sets later. When you are using more than one alias level, the catalog is chosen which has the most matching qualifiers to the data set being searched for or cataloged.

*General Considerations for Multilevel Aliases:* Before selecting an alias level, or a specific multilevel alias, consider the following:

1. User catalog names perform like aliases. For example, if you have a catalog USER.ICFUCAT1 with the alias search level set at 2, then all data sets beginning with USER.ICFUCAT1 are cataloged in this catalog.

2. When defining an SMS-managed data set with multiple components, like an SMS-managed VSAM key-sequenced data set, if a user specifies a component name which resolves to a *different* catalog than the data set itself, the definition fails.

3. When defining a VSAM cluster or a generation data group, if the name of the cluster or generation data group matches an existing alias or user catalog name, the definition fails with a "duplicate name" error. This is to prevent the data or index component of the VSAM data set, or a generation data set, from becoming inaccessible.

For example, consider a situation where two user catalogs, ICFUCAT1 and ICFUCAT2, each have an alias. The alias search level is set at 3. There are two aliases defined:

1. Alias A.B for catalog ICFUCAT1
2. Alias A.B.C for catalog ICFUCAT2

If you define an entry-sequenced data set named A.B.C without specifying the data component name:

1. The cluster entry is in ICFUCAT1
2. The generated data component name is A.B.C.DATA

Since A.B.C is an alias for ICFUCAT2, any references to A.B.C.DATA are oriented to ICFUCAT2. Since A.B.C is cataloged in ICFUCAT1, any request for the data component results in a "data set not found" error.

Also, if you have a generation data group named A.B.C, it points to ICFUCAT1. However, the names of the generation data sets are in the form of A.B.C.G*xxxx*V*yy*. Even though the generation data sets are actually cataloged in ICFUCAT1 because its base is in ICFUCAT1, later references to any generation data set results in orientation to ICFUCAT2 and a "data set not found" error.

This problem can be avoided if no VSAM data set or generation data group is defined with the same name as an existing alias or catalog. It is your responsibility to ensure that when you add an alias to the catalog, you do not cause existing data sets to become inaccessible.

***Procedure for Choosing a Multilevel Alias:*** You should take these steps before defining new aliases:

1. Do a LISTCAT LEVEL, using only the first qualifier of the new alias. For example, if the new alias is A.B.C, execute LISTCAT LEVEL(A). For aliases of only one level, the "first" qualifier is the alias name.

2. Check for any matches. If there are any matches and the listed data set does not currently reside in the catalog you are attempting to define the alias for, you might need to rename the data set or choose another alias if the data set becomes inaccessible.

For example, if you want to define the alias A.B for ICFUCAT2, use LISTCAT LEVEL(A) and analyze the output. In this example, A is an alias for ICFUCAT1 and A.B.C is an alias for ICFUCAT3:

```
CLUSTER ------ A.CLUSTER
      IN-CAT -- ICFUCAT1
DATA --------- A.DATA
      IN-CAT -- ICFUCAT1
CLUSTER ------ A.B.CLUSTER2
      IN-CAT -- ICFUCAT1
DATA --------- A.B.DATA2
      IN-CAT -- ICFUCAT1
CLUSTER ------ A.B.CLUSTER
      IN-CAT -- ICFUCAT1
DATA --------- A.B.DATA
      IN-CAT -- ICFUCAT1
CLUSTER ------ A.B.C.CLUSTER
      IN-CAT -- ICFUCAT3
DATA --------- A.B.C.DATA
      IN-CAT -- ICFUCAT3
INDEX -------- A.B.C.INDEX
      IN-CAT -- ICFUCAT3
ALIAS -------- A
      IN-CAT -- ICFMASTR
ALIAS -------- A.B.C
      IN-CAT -- ICFMASTR
```

Evaluation of the LISTCAT output:
The data sets A.CLUSTER and A.B.C.CLUSTER and their components will remain accessible. With an alias search level of 3, their aliases (A and A.B.C) continue to be oriented to the correct catalog.
However, A.B.CLUSTER2 and A.B.CLUSTER will become inaccessible if you define A.B as an alias for ICFUCAT2. These data sets are cataloged in ICFUCAT1, since they use the alias "A". After alias A.B is defined, searches for these data sets will be oriented to ICFUCAT2, and the data sets will not be found.

If you choose to rename a data set, the data and index components probably need to be renamed as well.

The following types of data sets might not be found by LISTCAT LEVEL:

1. Data sets defined with a catalog specified in the CATALOG parameter (bypassing the catalog search order) or with job or step catalogs.

2. Data sets whose data or index component names were defined by the user.

3. Data sets that became inaccessible because of the removal of an alias.

To identify data sets not cataloged according to catalog aliases, use the CATALOG parameter of LISTCAT, specifying the catalog containing the data set's entry. All user data sets should be cataloged according to catalog aliases. Otherwise, data can easily become lost, defeating the purpose of cataloging data.

## The Master Catalog

There is no structural difference between a master catalog and a user catalog. What makes a master catalog different is how it is used, and what data sets are cataloged in it. Each system has one active master catalog. The master catalog does not have to reside on the system residence volume. For performance, recovery, and reliability, we recommend that you only use ICF catalogs.

## Contents of the Master Catalog

The master catalog for a system must contain entries for all user catalogs, and their aliases, which the system uses. The only other data sets which you should catalog in the master catalog are the system, or "SYS1", data sets. These data sets must be cataloged in the master catalog for proper system initialization. Figure 4 shows the content and connections of a master catalog.
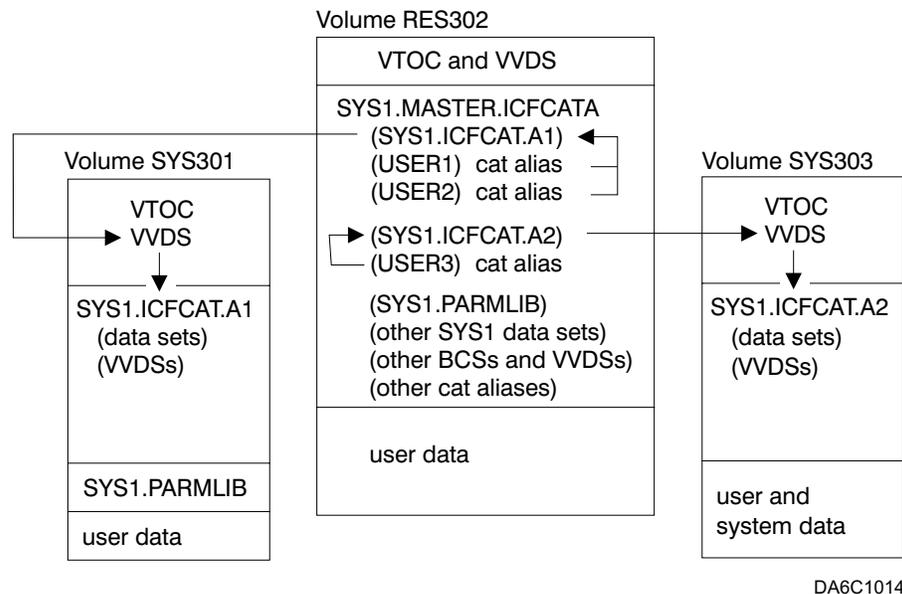


Figure 4. Relationship Between Master and User Catalogs. Parentheses indicate an entry in a catalog. Alias entries point to other entries in the same catalog. Arrows indicate the relationship between entries and the data sets they represent. The order of the entries in this example does not represent the order of entries in an actual catalog.

If your installation has multiple, interconnected systems, the master catalogs of each system can be connected to the master catalogs of each of the other systems. In other words, a master catalog on one system is a user catalog on the other systems.

You might want to combine SMS complexes into a single SMS complex and eliminate additional control data sets. To share the master catalog across an SMS complex see "Sharing a Master Catalog and IPL Volume" on page 26. If you are not running SMS, you can connect as many systems as is supported by the channel. RACF and appropriate alias naming conventions can prevent users on one system from cataloging data sets in the master catalog of another system. See Figure 3 on page 14 for an example of interconnected master catalogs.

For ease of backup and recovery of the master catalog, no user data sets should be cataloged in the master catalog. If you deny update access to the master catalog for most of your users, there is typically no update activity for the master catalog.

Each system must identify its master catalog.   You identify the master catalog in the SYSCAT*xx* member of SYS1.NUCLEUS, or the LOAD*xx* member of SYS1.PARMLIB. (See "Identifying the Master Catalog and Initial Configuration (SYSCATxx)" on page 31, and "Bypassing SYSCATxx with LOADxx" on page 33 for more information.)

## The Master Catalog During System Initialization

During a system initialization, the master catalog is read so that system data sets and catalogs can be located. Their catalog entries are placed in the cache that is selected by the user for the master catalog as they are read.

Catalog aliases are also read during system initialization, but they are placed in an alias table separate from the cache that is used for the master catalog.

Thus, if the master catalog only contains entries for system data sets, catalogs, and catalog aliases, the entire master catalog is in main storage by the completion of the system initialization.

## Creating and Using an Alternate Master Catalog

Because the master catalog is vital to the functioning of an z/OS system, you should create an alternate master catalog which can be used in a system initial program load (IPL) if the regular master catalog becomes damaged.

At minimum, an alternate master catalog contains entries for the system data sets necessary to IPL the system. After IPL, the original master catalog can be repaired or recovered, and the system can again be IPLed with the newly recovered master catalog.

The simplest procedure for creating an alternate master catalog is to use the access method services REPRO command to copy the master catalog into a defined new master catalog. After you have copied the master catalog into the newly defined master catalog, complete the following steps:

1. Use the newly defined catalogs as the master and the old master catalog as the alternate master catalog. This is necessary because the REPRO process changes the VVDS pointers to the output catalog. Although the system can still be IPLed with the old master, allowing you to recover the new master catalog if necessary, the new master should be used under normal circumstances.

2. Define the alternate master on a different volume than the volume of the original master. Otherwise, if the original master's volume is damaged, both the original master and alternate master are unavailable. If you desire volume IPL, initialize the new volume with IPL text, and copy the required system data sets to the

new volume. Allocate the new system data sets with a different high level qualifier than SYS1 (e.g. SYS2), then rename them after they are cataloged in the alternate master. This is necessary because some system data sets are always in use. The alternate master then has entries for the same data sets defined in the original master.

3. To manually create a minimum alternate master catalog without using REPRO, create the catalog and define all the system data sets in the alternate master. Then define new storage index and page data sets.

The newly defined catalog should then be used as the master catalog, and the old master should be used as the alternate master. The alternate master should be defined on a different volume than the volume of the original master: otherwise, if the original master's volume is damaged, both the original master and the alternate master are unavailable.

After the alternate is created, changes to the entries for the system data sets are only reflected in the master catalog which is in use. If you IPL the system with the new master, changes are not reflected in the old master (now the alternate master). For example, if a system data set is moved to a different volume, update the alternate master by recreating it or recataloging the data set.

After defining the new master and copying the original master into it, create a SYSCAT*xx* member in SYS1.NUCLEUS to identify the alternate master. To use the alternate master catalog, specify at IPL time the two-character identifier of the SYSCAT*xx* member which contains the entry identifying the alternate master catalog. You can also define the alternate master in a LOAD*xx* member in SYS1.PARMLIB.

If you want to maintain the SYSCATLG member as the member identifying your current master catalog, and the SYSCATLG member points to the old master catalog, copy the member to a different SYSCAT*xx* member. Then update the SYSCATLG member to point to the catalog you just created.

The following example shows the procedure for creating an alternate master catalog. The new master catalog created is SYS1.ICFCAT.NEWMASTR. The old master catalog SYS1.ICFCAT.MASTER is used as the alternate master catalog, and is identified in the SYSCATAL member of SYS1.NUCLEUS. Both the new master and the old master use a multilevel alias search level of 1, no SYS% conversion, and the default number of CAS tasks.

```
//STEP01   EXEC  PGM=IDCAMS
//SYSPRINT DD  SYSOUT=A
//SYSIN    DD  *
   /***********************************************************/
   /* DEFINE THE NEW MASTER CATALOG                          */
   /* VOLUME ALTVOL DOES NOT CONTAIN THE ORIGINAL MASTER CATALOG*/
   /***********************************************************/
      DEFINE MASTERCATALOG -
            ( NAME(SYS1.ICFCAT.NEWMASTR) -
              CYLINDERS (5 1)  -
              VOLUME (ALTVOL) -
              ICFCATALOG)
/*
//STEP02   EXEC  PGM=IDCAMS
//ALTERV    DD  UNIT=3390,VOL=SER=ALTVOL,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   /***************************************************************/
   /* COPY OLD MASTER CATALOG TO THE NEW MASTER CATALOG          */
   /***************************************************************/
      REPRO INDATASET(SYS1.ICFCAT.MASTER) -
            OUTDATASET(SYS1.ICFCAT.NEWMASTR)
/*
//STEP03   EXEC PGM=IEBGENER
//************************************************************
//* COPY OLD SYS1.NUCLEUS(SYSCATLG) TO SYS1.NUCLEUS(SYSCATAL) *
//* (USE OLD MASTER AS THE ALTERNATE MASTER, NEW MASTER AS THE*
//*  DEFAULT MASTER)                                          *
//************************************************************
//SYSPRINT DD SYSOUT=A
//SYSUT1   DD DSN=SYS1.NUCLEUS(SYSCATLG),DISP=OLD
//SYSUT2   DD DSN=SYS1.NUCLEUS(SYSCATAL),DISP=OLD
//SYSIN    DD DUMMY
//STEP04   EXEC PGM=IEBGENER
```

*Figure 5. Creating an Alternate Master Catalog (Part 1 of 2)*

```
//************************************************************
//* REPLACE OLD SYS1.NUCLEUS(SYSCATLG) MEMBER WITH POINTER TO  *
//* NEW MASTER CATALOG.                                       *
//************************************************************
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  *
ALTVOL11  SYS1.ICFCAT.NEWMASTR
/*
//SYSUT2   DD  DSN=SYS1.NUCLEUS(SYSCATLG),DISP=OLD
//SYSIN    DD  DUMMY
```

*Figure 5. Creating an Alternate Master Catalog (Part 2 of 2)*

## Sharing a Master Catalog and IPL Volume

With the introduction of MVS/SP 5.1.0, an installation that has multiple MVS images can share a master catalog and share an IPL volume among multiple MVS images. The system data sets, SYS1.LOGREC and SYS1.STGINDEX are no longer fixed named and unable to be shared. They can now be shared and specified by the installation. In addition, a system symbolic, &SYSNAME, was introduced and can be used as part of data set name specifications for some parameters in PARMLIB. When you use &SYSNAME, the data set name specification becomes flexible and you do not need a separate parameter specification for each system in the sysplex.

For example, LOGREC=SYS1.LOGREC.&SYSNAME. The symbolic name, &SYSNAME, can also be used in other PARMLIB parameter specifications. You can use &SYSNAME for IEASYSxx parameters VIODSN=, PAGE=, SWAP=, DUPLEX=, and NONVIO=. You can use &SYSNAME for SMFPRMxx parameters DSNAME= and SID=.

## Using Symbolic References for Aliases

When sharing a master catalog, it might be desirable for an alias to reference a different data set or catalog, depending on the system it is accessed from. You can now define an alias for a catalog or a non-VSAM data set where the catalog name or non-VSAM data set name contains system symbols. This name is resolved at the time the alias entry is referenced, rather than when it is defined.

Aliases of data sets may not be used prior to the start of Master Scheduler Initialization. This is approximately the time that the system switches to fullscreen console mode. Thus, cataloged data set names contained in system initialization members that are read and processed prior to this time may NOT be specified as aliases. The actual data set names must be provided.

This capability simplifies staged introduction of products on members of a sysplex. Now a single entry in the shared master catalog can refer to different physical data sets on each member of the sysplex at the same time.

For more information, see the DEFINE ALIAS description of the SYMBOLICRELATE keyword in *z/OS DFSMS Access Method Services.*

## Catalog Performance

This section is intended to help you to do tuning of DFSMS. This section documents information which is tuning information provided by DFSMS.

Performance should not be your main consideration when you define catalogs. It is more important to create a catalog configuration which allows easy recovery of damaged catalogs with the least amount of system disruption. However, there are several options you can choose to improve catalog performance without affecting the recoverability of a catalog.

## Factors Affecting Catalog Performance

The main factors affecting catalog performance are the amount of I/O required for the catalog and the subsequent amount of time it takes to perform the I/O. These factors can be reduced by caching catalogs in special caches used only by catalogs.

If the master catalog only contains entries for catalogs, catalog aliases, and system data sets, the entire master catalog is read into main storage during system initialization. Because the master catalog, if properly used, is rarely updated, the performance of the master catalog is not appreciably affected by I/O requirements.

Performance can also be enhanced by specifying certain options when defining a catalog. These performance attributes are discussed in "Choosing Options to Adjust Catalog Performance" on page 46, and "Setting the Catalog Control Interval and Area Size" on page 42.

Sharing a catalog between systems also affects performance. See "The Effect of Sharing Catalogs on Cache Usage" on page 30.

Finally, eliminating JOBCAT and STEPCAT DD statements from job streams improves job and catalog performance, especially if two or more catalogs are concatenated on the DD statement. Catalog searches should be performed using catalog aliases.

In summary, catalog performance is mainly enhanced by caching the catalog. If a catalog is cached, the reduced I/O to the catalog reduces the effect of other factors on catalog performance. Otherwise, considerations affecting catalog recovery should take precedence over performance considerations.

# Eliminating JOBCAT and STEPCAT DD Statements from Job Streams

The use of JOBCAT and STEPCAT DD statements impacts the performance of a job, since job and step catalogs are searched before other catalogs. If two or more catalogs are concatenated on these DD statements, the catalogs are searched in the order of the concatenation. If the data set being located is not cataloged in the first catalog in the concatenation, more catalogs are searched than necessary, so that the catalog search takes longer than necessary.

In contrast, if all data sets are located according to catalog aliases, only one catalog is searched for each data set. Thus, the catalog search takes the minimum amount of time since catalogs are not searched needlessly.

Concatenated JOBCAT and STEPCAT DD statements can cause serious performance degradation and unpredictable results if the number of catalogs specified causes the maximum number of opened catalogs limit to be exceeded. "Data set not found" could be a result.

During system initialization, the maximum number of catalogs which can be open simultaneously is set at 1024. This number can be changed using the MODIFY CATALOG,CATMAX command. The maximum number must be greater than or equal to the maximum number of catalogs used in a concatenated JOBCAT or STEPCAT DD statement.

Excessive opening and closing of catalogs increases as more catalogs are concatenated on the STEPCAT DD statement. Not only is the performance of the job degraded, the performance of all jobs on the system is affected. When the step terminates, catalogs specified on the STEPCAT DD statement which were not open before the beginning of the step are closed.

If these catalogs are using the catalog data space cache, additional performance degradation occurs. When a catalog using the catalog data space cache is closed, the space it used in the catalog data space is freed. Because none of its records are already in cache when it is reopened, no catalog requests are immediately satisfied from the cache.

To prevent these potential performance problems, eliminate all JOBCAT and STEPCAT DD statements from the job streams at your installation, and have every catalog request satisfied by catalog aliases.

# Caching Catalogs

The simplest method of improving catalog performance is to use cache to maintain catalog records within main storage or data space. Using cache reduces the I/O required to read records from catalogs on DASD.

Two kinds of cache are available exclusively for catalogs. The in-storage catalog (ISC) cache is contained within the catalog address space (CAS) in main storage. The catalog data space cache (CDSC) is separate from CAS and uses the MVS VLF component which stores the cached records in a dataspace. Both types of cache are optional, and each can be cancelled and restarted without an IPL.

Although you can use both types of catalog cache, you cannot cache a single catalog in both types of cache simultaneously. You must decide which catalogs benefit the most from each type of cache.

Catalog records are cached in the ISC or CDSC under the following conditions:
- For master catalogs, all records accessed sequentially or by key are cached except for alias records. Alias records are kept in a separate table in main storage.
- For user catalogs, only records accessed by key are cached.
- For each catalog, the records are cached in the CDSC if you have indicated the catalog is to use CDSC. Otherwise, the records are cached in the ISC, unless you have stopped the ISC for the catalog. If you stop both the CDSC and the ISC for a catalog, then records are not cached.

See "Starting and Stopping the Catalog Cache for a Catalog" on page 121 for information on using the MODIFY CATALOG command to manipulate the caches.

## In-Storage Catalog Cache
The in-storage catalog cache resides in main storage within the catalog address space. It is the default catalog cache. Each user catalog cached in ISC is given a fixed amount of space for cached records. When a user catalog uses its allotted space in the ISC, the least recently used record is removed from the ISC to make room for the new entry.

Catalogs which are not frequently updated use the ISC most effectively. The master catalog is ideally suited for ISC, for example, because master catalog records are infrequently updated, unless the master catalog is shared with other systems. The performance of the ISC is affected if the catalog is shared with another system. See "The Effect of Sharing Catalogs on Cache Usage" on page 30 for more information.

Master catalogs, unlike user catalogs, are not limited to a set amount of storage. All eligible records in the master catalog are cached in the ISC as they are read. Thus, you should keep the number of entries in the master catalog to a minimum, so that the ISC for the master does not use an excessive amount of main storage.

Since ISC is the default catalog cache, catalogs are cached in the ISC unless you specify that the catalog is to use CDSC, or unless you use the MODIFY CATALOG operator command to remove the catalog from the ISC.

## Catalog Data Space Cache
The catalog data space cache (CDSC) resides in a data space which you define with the COFVLF*xx* member of SYS1.PARMLIB. The CDSC uses the virtual

lookaside facility (VLF), which can be started using the START VLF operator command. See "Defining the Catalog Data Space Cache (COFVLFxx)" on page 34 for information on defining the CDSC.

You can add catalogs to the CDSC only by editing the COFVLF*xx* member to specify the catalogs, stopping VLF, then starting VLF. Because this releases the existing CDSC, catalog performance might be degraded for a while.

The CDSC can concurrently be used for any catalog which is not using the ISC. A single catalog cannot use both the CDSC and the ISC. Unlike the ISC, catalogs cached in the CDSC are not limited to a specific amount of storage. A catalog caches records until no space is left in the data space cache. Once the data space cache is full, the space occupied by the record least used is removed to make room for new records.

See "Monitoring the Catalog Address Space" on page 107 for information on monitoring the performance of the CDSC.

## The Effect of Sharing Catalogs on Cache Usage

If a catalog is defined with share options (3 4), and if it resides on a shared device, catalog management considers the catalog a shared catalog. A catalog is considered a shared catalog if it meets both of these conditions, even if it is not actually being shared among systems. Before each physical access to a shared catalog, special checking is performed to ensure that the ISC or CDSC contains current information. Checking also ensures that the access method control blocks for the catalog are updated in the event the catalog has been extended, or otherwise altered from another system. This checking maintains data integrity. It also affects performance because the VVR for a shared catalog must be read before using the ISC or CDSC version of the BCS record.

**Recommendation:** To avoid catalog corruption, define a catalog volume on a shared UCB and set catalog share options to (3 4) on all systems sharing a catalog.

A single catalog request can involve many physical references to the catalog, so the effect of this additional access to the VVDS can be significant. The volume containing the VVDS will be reserved, and I/O will be performed. If the catalog has been defined to support Enhanced Catalog Sharing (ECS) by the ECSHARING attribute, most of this overhead will be eliminated.

Changes to shared catalogs are handled differently depending on whether the catalog uses the ISC or the CDSC.

If a catalog uses the ISC and a sharing system updates a record (any record, even if the record is not cached in this system's ISC), catalog management releases the *entire* ISC for the catalog and creates a new ISC for the catalog. Individual records changed by a sharing system are not identified and updated for ISC catalogs.

The CDSC, however, can identify individual records which a sharing system has updated. Thus, when a sharing system updates a record, the CDSC space used by the catalog is not necessarily released. CDSC space for a catalog is only "invalidated" (marked unusable and given back to the CDSC as "free space" which any catalog can use) if so many changes were made by a sharing system that catalog management could not maintain a record of all the changes. Otherwise, all changes made by the sharing system can be made to the CDSC record by record.

If a catalog is not really shared with another system, move the catalog to an unshared device or alter its share options to (3 3). To prevent potential catalog damage, never place a catalog with share options (3 3) on a shared device.

# Defining the Catalog Configuration

The system data sets SYS1.NUCLEUS and SYS1.PARMLIB contain members used to define portions of your catalog configuration.

When you IPL your z/OS system, you must identify the master catalog to z/OS. This can be done either through the SYSCAT*xx* member of SYS1.NUCLEUS or the LOAD*xx* member of SYS1.PARMLIB.

If you use the SYSCAT*xx* member, z/OS issues the following message asking you to identify which member you are using. This is done during the nucleus initialization program (NIP) time. You respond with the last two characters in the SYSCAT*xx* member name (the *xx* value):

```
IEA347A SPECIFY MASTER CATALOG PARAMETER
```

If you enter a blank line in response to this message, z/OS uses "LG" as the parameter, identifying SYSCATLG.

If you use the LOAD*xx* member to identify the master catalog, the system uses the master catalog specified in the member, and the master catalog message is not issued.

MVS also uses information you define in the COFVLF*xx* member of SYS1.PARMLIB to identify the catalogs which are candidates for the catalog data space cache (or VLF cache). Catalogs not identified in the COFVLF*xx* member are eligible for the standard in-storage catalog cache.

Another member of SYS1.PARMLIB, SMFPRM*xx*, is used to identify the record types that the system management facilities records. SMF can be used to record changes to catalogs.

# Identifying the Master Catalog and Initial Configuration (SYSCATxx)

| Attributes Defined in SYS1.NUCLEUS(SYSCATxx) |
|---|
| Master catalog's volume |
| Master catalog type |
| SYS% facility on or off |
| Multilevel alias search level |
| Catalog address space service task lower limit |
| Master catalog name |
| Tape volume catalog high level qualifier |

Once you have defined your master catalog, and have cataloged the necessary system data sets in it, you must identify the master catalog to the system to use it as a master catalog. The SYSCAT*xx* member of SYS1.NUCLEUS contains the information for identifying the master catalog to MVS.

If you use the LOAD*xx* member to identify the master catalog, the system uses the master catalog specified in the member, and the master catalog message is not issued.

You can have multiple SYSCAT*xx* members. The default name is SYSCATLG. Each copy of the SYSCAT*xx* member can identify different master catalogs.

For more information on using the SYSCAT*xx* member, see *z/OS MVS System Data Set Definition* .

The following is the format of the SYSCAT*xx* member of SYS1.NUCLEUS (the member can contain only one record):

| volser | catalog type | alias level | CAS task low limit | catalog name | tape volume catalog high level qualifier |
|--------|--------------|-------------|--------------------|--------------|------------------------------------------|

(Optional)

Bytes: 1       7        8     9         11     55

where:

*volser*  is the volume serial number of the master catalog's volume. This must be in the first 6 bytes of the record.

*catalog type*
is the catalog type and SYS% facility default, where:

blank or **0**
Indicates a VSAM master catalog. In this case, leave bytes 8, 9, and 10 blank. Only the volume serial number and the catalog name can be specified for a VSAM master catalog.

**1**  indicates an ICF catalog, with the SYS% facility turned *off*,

**2**  indicates an ICF catalog with the SYS% facility turned *on*.

This value must be defined in the seventh column of the record. The setting for the SYS% conversion facility can be changed after IPL for a current session with the MODIFY CATALOG,SYS% operator command.

*alias level*
specifies the multilevel alias search level. The default is 1, and the maximum is 4.

This value must be defined in the eighth column of the record. If you want the default value of 1, either specify 1 or leave the eighth column blank.

This value can later be changed for a current session with the MODIFY CATALOG,ALIASLEVEL operator command.

*CAS task low limit*
specifies the catalog address space service task lower limit, in hexadecimal. The value can range from hexadecimal 18 to hexadecimal B4, but the default is hexadecimal 3C. To specify hexadecimal B4, you can enter C'B4' or X'C2F4'.

This value must be defined in the ninth and tenth columns. Specify the default value (either explicitly, or by leaving these two columns blank). If the catalog address space needs more services tasks, it creates them.

*catalog name*
> specifies the name of the master catalog. The name can be up to 44 characters.
>
> To change to a different master catalog, you have to IPL the system and specify a different SYSCAT*xx* member.

*tape volume catalog high level qualifier (optional, otherwise blank)*
> specifies the first name qualifier of all volume catalogs (volcats) in the system. The value is specified as 1 to 8 characters.   See "Defining Names for a Tape Volume Catalog" on page 50. Leave this blank if you have no tape volume catalog member.

The following is a sample step to create the SYS1.NUCLEUS member SYSCATLG. This job describes an ICF master catalog named SYS1.MASTERA.ICFCAT on volume SYSRES. The SYS% to SYS1 conversion facility is turned off. The multilevel alias search level is set at one. The catalog address space service task lower limit default of X'3C' is used by leaving columns 9 and 10 blank.

```
//SYSCAT   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN    DD DUMMY
//SYSUT2   DD DSN=SYS1.NUCLEUS(SYSCATLG),DISP=SHR,DCB=(RECFM=U)
//SYSUT1   DD *
SYSRES11  SYS1.MASTERA.ICFCAT
/*
```

# Bypassing SYSCATxx with LOADxx

| **Attributes Defined in SYS1.PARMLIB(LOADxx)** |
| --- |
| Same attributes as SYS1.NUCLEUS(SYSCAT*xx*) |

The LOAD*xx* member of SYS1.PARMLIB can be used to specify the master catalog and other information needed to initialize a system. If the master catalog is specified in this member, the operator is not asked to specify a SYSCAT*xx* member during system initialization.

To identify the master catalog in the LOAD*xx* member, use the SYSCAT statement. The following is the format of the SYSCAT statement for the LOAD*xx* member:

| SYSCAT | volser | catalog type | alias level | CAS task low limit | catalog name | tape volume catalog high level qualifier |
| --- | --- | --- | --- | --- | --- | --- |

```
Bytes:
1        10      16       17    18         20        64
```

The information following the SYSCAT keyword is the same as that in the SYSCAT*xx* member of SYS1.NUCLEUS. (See "Identifying the Master Catalog and Initial Configuration (SYSCATxx)" on page 31 for an explanation of the values.) The high level qualifier of the tape volume catalog is specified with 1 to 8 characters or the default is SYS1. When the system is initialized with SYSP=*xx*, the tape volume catalog high level qualifier specified in the LOAD*xx* member is activated.

The following SYSCAT statement in LOAD*xx* replaces the SYSCATLG member created in "Identifying the Master Catalog and Initial Configuration (SYSCATxx)" on page 31:

```
SYSCAT   SYSRES11 SYS1.MASTERA.ICFCAT
```

For complete information on the format and use of the LOAD*xx* member of SYS1.PARMLIB, see *z/OS MVS Initialization and Tuning Reference*.

# Defining the Catalog Data Space Cache (COFVLFxx)

| **Attributes Defined in SYS1.PARMLIB(COFVLFxx)** |
|---|
| Catalogs eligible for the catalog data space cache<br>Amount of storage used for the catalog data<br>space cache |

The COFVLF*xx* member of SYS1.PARMLIB determines which catalogs are eligible for the catalog data space (VLF) cache. Tape volume catalogs can be in the VLF cache. The cache is not actually started until you enter the START VLF operator command, which is explained in *z/OS MVS System Commands* . For an explanation of the catalog data space cache, and how to determine which catalogs should be eligible for data space caching, see "Caching Catalogs" on page 29.

Although the COFVLF*xx* member can also be used by other components, the discussion here focuses strictly on its use with catalogs. For more information on this SYS1.PARMLIB member, see *z/OS MVS Initialization and Tuning Reference*.

Besides specifying the catalogs eligible for the catalog data space cache, you can also specify the amount of storage that you will allow VLF to use to cache catalog records.

The syntax of the CLASS statement for defining the catalog data space cache is:

| **CLASS** | **NAME(IGGCAS)** |
|---|---|
| | **EMAJ***(catname)* |
| | **[EMAJ***(catname)*] |
| | **[...]** |
| | **MAXVIRT({ 4096|***size***})** |

where:

**NAME(IGGCAS)**
　　specifies the class name for the catalog data space cache.

**EMAJ(***catname***)**
　　specifies the name of an ICF catalog eligible for catalog data space caching.

　　You can include any number of EMAJ parameters on the CLASS statement, but each EMAJ parameter must specify one, and only one, catalog.

　　Catalogs cached in the catalog data space cache cannot be simultaneously cached in the in-storage catalog. For a complete discussion of caching catalogs, see "Caching Catalogs" on page 29.

**MAXVIRT({4096|***size***})**
　　specifies the maximum virtual storage that VLF can use to cache catalog records. The decimal value is multiplied by 4096 (4K blocks). The minimum

value for size is 256 decimal (256 equates to one megabyte). The maximum value for size is 524288 decimal. If you specify a decimal value that is not in the valid range, VLF defaults to 4096 decimal. To determine the optimum size for MAXVIRT, see"Evaluating Catalog Data Space Cache Performance" on page 110. You can also refer to the *z/OS MVS Initialization and Tuning Reference*.

For example, the following CLASS statement identifies four catalogs as eligible for catalog data space caching:

```
CLASS   NAME(IGGCAS)
        EMAJ(USER.ICFCAT.PROJECT1)
        EMAJ(USER.ICFCAT.PROJECT2)
        EMAJ(USER.ICFCAT.PRODUCT1)
        EMAJ(USER.ICFCAT.PRODTEST)
        MAXVIRT(256)
```

You can monitor and evaluate the performance of the catalog data space cache with the information supplied by the MODIFY CATALOG,REPORT,VLF operator command. See "Monitoring the Catalog Address Space" on page 107 for more information.

# Recording SMF Records for Catalog Events (SMFPRMxx)

You can use the system management facilities to record certain catalog events. These records can be used to keep track of catalog activity, and can be used during catalog recovery. These records are also valuable in diagnosing catalog problems, and should always be recorded.

With complete SMF information, you can determine which backup copy of a catalog is the most recent, and you can use the Integrated Catalog Facility Forward Recovery Utility to update a newly recovered catalog. You can also develop reporting applications to analyze or track catalog usage.

Table 3 lists the SMF record types used with the ICF. Define the SMFPRM*xx* member of SYS1.PARMLIB so that these record types are recorded. For complete information about defining this member, see *z/OS MVS Initialization and Tuning Reference*.

You can use the IFASMFI6 macro to map the fields of these records. For more information about using SMF data and the IFASMFI6 macro, and about the contents of these record types, see *z/OS MVS System Management Facilities (SMF)*.

*Table 3. SMF Record Types used with the ICF*

| Type | Description |
|------|-------------|
| 36 | Records successful exports of ICF catalogs. Contains the date and time of the export, and information needed for importing the backup copy. Written on successful completion of EXPORT. |
| 60 | Records any changes to VVDS records, both VVRs and NVRs. Written when a VVDS record is inserted, updated, or deleted. |
| 61 | Records changes to the BCS during DEFINE processing. Written when a record is inserted or updated. |
| 65 | Records changes to the BCS during DELETE processing. Written when a record is deleted or updated. |
| 66 | Records changes to the BCS during ALTER processing. Written when a record is updated, inserted, or deleted. |

# Using Enhanced Catalog Sharing Mode

ECS mode provides a substantial performance benefit for catalogs that are shared between one or more systems in a sysplex.

To use ECS mode, you must include the ECS structure in the Coupling Facility Resource Manager (CFRM) policy. You must also define (or alter) one or more catalogs to set the ECSHARING attribute.

## Defining the CFRM Policy for Enhanced Catalog Sharing

The following ECS structure information should be included in the IXCMIAPU job that formats the couple data set containing the Coupling Facility Resource Manager (CFRM) policy:

**Name:**
SYSIGGCAS_ECS

**Size:** Use the following parameters to determine the maximum structure size. See the *ES/9000 Processor Resource/Systems Manager Planning Guide* for the algorithms that use these parameters to estimate the size of the cache structure. You can also use the CFSIZER to determine how large the structure should be and to generate the JCL. CFSIZER is available at the following website: *www.ibm.com/s390/pso/*

* Total directory entry count (TDEC): The expected total number of ECS-active catalogs in the sysplex at any one time plus 25%. The 25% is a recommended buffer so that the structure never becomes full. If it does, then the CF default algorithm of reclaiming the least-recently-used entries will take effect. If the structure ever does become full, either catalogs will not be activated or a performance degradation will be experienced for some catalogs because of the overhead involved in rebuilding entries that were reclaimed. The maximum number of ECS-active catalogs that are supported on any one system is 1024.
* Total data area element count (TDAEC): Same as TDEC
* Maximum storage class (MSC): 1
* Maximum castout class (MCC): 1
* Maximum data area size (MDAS): 1
* Data area element characteristic (DAEX): 4
* Adjunct Assignment Indicator (AAI): 0 (zero)
* Directory portion of directory-to-data ratio (R_de): 1
* Data portion of directory-to-data ratio (R_data): 1

The size of the structure must be supplied by the installation in the CFRM policy. (The ECS facility does not specify a size.)

**Initial Size:**
Since ECS supports dynamic structure size alteration, an initial size (with a buffer smaller than the recommended 25% for the maximum size described above) can be specified in the CFRM policy. See the chapter entitled ″Managing Coupling Facility Resources″ in *z/OS MVS Setting Up a Sysplex* for information on specifying an initial size.

**Preference List:**
Determined by the installation. The ECS cache structure does not require a nonvolatile CF.

**Exclusion List:**
Determined by the installation.

The ECS cache structure supports the rebuild and alter functions.

## Enabling a Catalog for ECS Mode

A new attribute, ECSHARING, is available on the IDCAMS DEFINE and ALTER commands. Setting this attribute will make a catalog eligible for sharing using the ECS protocol, as opposed to the VVDS protocol. However, the system will not actually use the ECS protocol unless:

- There is an active connection to the ECS cache structure AND
- The ECS mode has been activated by the MODIFY CATALOG, ECSHR(AUTOADD) command.

You can alter the ECSHARING attribute of a catalog at any time. If you remove the attribute and the catalog is currently using ECS mode, it will be converted back to VVDS mode on all systems that are sharing it. This is convenient if you find that the catalog must be accessed by a system that does not support the ECS protocol. A catalog cannot be shared using both the ECS and VVDS protocols at the same time. Therefore, it is necessary to stop ECS mode for a catalog if access is needed by a system that does not support the ECS protocol. This can be temporarily accomplished by using the MODIFY CATALOG ECSHR(REMOVE,catname) command. This does not remove the ECSHARING attribute, but does remove the catalog from ECS mode on all systems. It can only be returned to ECS mode by issuing a MODIFY CATALOG ECSHR(ENABLE,catname) command.

## Restrictions on ECS Mode Usage

There are several restrictions on using ECS mode:

- ECS and VVDS protocols cannot be used simultaneously for a catalog. This is enforced by the catalog address space. If you attempt to use a catalog that is currently being shared by ECS mode from a system without ECS support enabled, the associated catalog request will fail. The return code is RC228 and the reason code is RSN26.
- No more than 1024 catalogs can currently be shared using ECS from a single system.
- All systems sharing the catalog in ECS mode must have connectivity to the same Coupling Facility, and must be in the same global resource serialization (GRS) complex.

## Activating ECS

To activate ECS, perform the following steps:

1. Define the ECS structure in the CFRM policy and activate the policy. This action should connect all ECS-eligible systems to the ECS structure.
2. Specify the ECSHARING attribute for the desired catalogs so that they are ECS-eligible.
3. Issue the MODIFY CATALOG,ECSHR(AUTOADD) command on one system to enable AutoAdd throughout the sysplex. This will cause all catalogs that are eligible to be automatically activated on their next reference.

   Restriction: AutoAdd should never be enabled if all participating systems are not connected because once AutoAdd is enabled, it will automatically add catalogs to the ECS structure on the next reference on each system. For example, if System A is connected and System B is not connected and AutoAdd is enabled, catalogs will automatically be added on System A but not on System B. Therefore, two systems would try to share the same catalog using different sharing protocols (System A uses ECS; System B uses VVDS sharing). Because data integrity would be compromised in this situation, ECS detects and prevents this situation by rejecting

System B's catalog requests with RC228 RSN26. If this situation occurs, issue the MODIFY CATALOG, ECSHR(DISCONNECT) command on all connected systems.

4. If a catalog is eligible (has the ECSHARING attribute) but still does not activate automatically and the MODIFY CATALOG, ECSHR(STATUS) command shows a status of Inact(NonECSAcc), then use the MODIFY CATALOG,ECSHR(ENABLE,catname) command to activate the catalog.

## Operational Considerations

This section includes information about the following operational considerations:

* Connecting to the structure
* The AutoAdd function
* Quiecsing ECS activity

*Connecting to the Structure:* The Catalog Address Space (CAS) automatically connects to the ECS structure in the coupling facility as soon as possible during its initialization. Operator message IEC377I is issued to indicate the success or failure of this action. (You will normally see this message two times during the IPL process.) If your CFRM policy does not have the ECS structure defined, you will receive the ″Not Connected″ variation of the IEC377I message; this is an informative message and can be ignored. If at any time the connection is lost, you can connect (or disconnect) to (or from) the structure using the MODIFY CATALOG command.

*The AutoAdd Function:* AutoAdd adds ECS-eligible catalogs to the ECS structure on the next reference to the catalog. When CAS connects to the ECS structure, it checks to see if AutoAdd is already enabled for the sysplex. If so, CAS checks to see if AutoAdd is enabled for this system as well. If AutoAdd is not enabled, ECS will not automatically enable it. The initial activation of AutoAdd must be performed manually by an informed operator who knows that there are no ECS-ineligible systems that will be blocked from catalog access by the activation of the ECS AutoAdd function. AutoAdd remains active until one of the following events occurs:

* ECS activity is quiesced in the sysplex. (ECS activity is quiesced during a rebuild but reactivates AutoAdd at the completion of the rebuild unless a rebuild error occurs.) See the ″Quiescing ECS Activity″ section below.
* All systems in the sysplex are down at the same time, such as during a power outage.
* Only one system is ECS-active and a CAS restart or an IPL is performed.

*Quiecsing ECS Activity:* ECS quiesces activity by turning off the AutoAdd function and removing all ECS-active catalogs from the ECS structure. All catalogs then revert to VVDS sharing mode. ECS activity is quiesced in a sysplex when one of the following events occurs:

* An ECS-active system normally disconnects from the ECS structure. A normal disconnect means that the system issuing the disconnect continues to function and make catalog requests. In this situation, ECS activity is quiesced to ensure that catalog requests on the disconnected system are not rejected because catalogs are ECS-active on other systems. Both CAS restarts and IPLs are considered abnormal disconnects and do not result in quiescing ECS activity. A normal disconnect is initiated by the MODIFY CATALOG,ECSHR(DISCONNECT) command.
* A rebuild of the ECS structure is initiated. This can be done by an operator, another component, or by ECS when it is informed of an ECS structure failure or loss of CF connection. ECS activity is quiesced for the life of the rebuild. Once

the rebuild has completed without error, ECS restores the ECS state at the time the rebuild was initiated; that is, if AutoAdd was on when the rebuild was initiated, AutoAdd is reactivated and all eligible catalogs are re-enabled automatically by ECS. If AutoAdd was not on when the rebuild was initiated, AutoAdd will not be activated at the completion of the rebuild.

**Recommendation:** In order to minimize catalog availability problems and operator intervention, do not set the ECSHARING attribute for a catalog unless all sharing systems will be ECS-active.

## RACF Considerations

The security administrator can define RACF profiles to control the use of coupling facility structures. If an IXLSTR.structure_name facility class is defined, then a PERMIT specifying the RACF user ID assigned to Catalog must be specified in this facility class definition. Refer to the chapter on "Connection Services" in the *z/OS MVS Programming: Sysplex Services Guide* for more detailed information about authorizing coupling facility requests.

# Chapter 3. Defining Catalogs

You create a catalog by defining its structure with the access method services DEFINE USERCATALOG command. The ICFCATALOG parameter is the default. DEFINE USERCATALOG VOLCATALOG defines a tape volume catalog that only contains tape library and tape volume entries. Access method services can also be used to define catalog aliases (with DEFINE ALIAS), and VVDSs (with DEFINE CLUSTER). Alias is not supported for tape volume catalogs.

Before you define a catalog, determine how large you want the catalog to be, and on which volume you want to place it. Also, determine the appropriate performance attributes for the catalog. All of these subjects are covered in this chapter.

See "Setting the Catalog Control Interval and Area Size" on page 42 for information on choosing an appropriate control interval size for the index.

## Determining Catalog Size

Ideally, a catalog should be defined with enough space so that it does not grow into secondary extents. Excessive secondary extents can decrease catalog performance. You only need to consider reorganizing the catalog to eliminate secondary extents if their number becomes excessive.

A catalog can have up to 123 extents but can only occupy space on a single volume.

Before defining a catalog, carefully consider how that catalog will be used. Estimate the number of data sets and OAM object collections that will be defined in the catalog. The size of the catalog is directly affected by the number of entries in the catalog, and by whether you are using the Storage Management Subsystem.

## Assigning Space to a Catalog

You can specify the required space for a catalog in kilobytes, megabytes, tracks, cylinders, or records. This value can be specified directly on the DEFINE USERCATALOG command, or indirectly, through an SMS data class defined for catalogs that contains a space attribute. You can specify the appropriate data class on the DATACLASS parameter, or you can allow the data class ACS routine to assign the appropriate data class.

If you specify space in records, the amount of space is determined by the average record size. This is specified on the RECORDSIZE parameter, and the default is 4086. In general, you should not change the default values for average and maximum record size.

You must always specify a space parameter at the USERCATALOG (cluster) level. Space is allocated to the BCS according to the following rules:

1. If space is only defined as a subparameter of USERCATALOG, the space specified is assigned to the data component of the catalog. Additional space is allocated to the index according to the size of the space request.
2. If space is specified as subparameters of USERCATALOG and DATA, then the data component is assigned the requested space and the index is allocated space according to the space request for DATA. The USERCATALOG space request is ignored.

3. If space is specified as subparameters of USERCATALOG, DATA, and INDEX, then the data and index components are assigned the requested space. The USERCATALOG space request is ignored. If INDEX space is specified, space must also be specified for DATA.

If you are defining the BCS on a volume which *does not* contain a VVDS, a VVDS is also defined and allocated with the BCS. This "implicitly defined" VVDS is allocated with primary and secondary space of 10 tracks. If you want to define a VVDS with a space allocation different from the default, you must define a VVDS on the volume (using DEFINE CLUSTER) before you define the BCS.

## Setting the Catalog Control Interval and Area Size

You can specify the control interval sizes, or the catalog can be allowed to select the CISIZE. The BCS is a VSAM key-sequenced data set; therefore, the standard control interval and control area calculations are used.

To set a specific control interval size (CISIZE), use one of the following guidelines:
- From 512 bytes to 8KB, CISIZE must be specified in multiples of 512 bytes
- From 8 kilobytes to 32 kilobytes, CISIZE must be specified in multiples of 2 kilobytes.

Selecting a 4096 or the 8192 byte control interval size provides an acceptable compromise between minimizing data transfer time and reduces the occurrence of a record that spans a control interval. The resulting values for the catalog should be the same as for a key-sequenced data set with spanned records.

Additional considerations include:
- For data records less than 1 kilobyte, specify a smaller control interval size (less than 32 768) for the data component. Most of the processing is random and little benefit is gained from larger control interval sizes.
- Large or multivolume data records (catalog records larger than the CISIZE) require a second read to obtain the record and this has a direct impact on performance. Specifying a larger control interval size can provide improved performance for VSAM-key sequenced data sets with many volumes (or alternate data indices), or Generation Data Groups containing large, multivolume data sets.

For more information on CISIZE considerations, see the IBM publication titled *z/OS DFSMS: Using Data Sets*

The control area size for the data component is the smaller of the primary allocation quantity, secondary allocation quantity, or one cylinder.

The control area size is never smaller than a track or greater than one cylinder. It should be large enough to contain a maximum-length record; the default maximum record length for the BCS (a spanned record data set) is 32 400 bytes.

**Recommendation:** The maximum record size can be specified in the RECORDSIZE parameter, but we recommend that you do not change it.

To optimize catalog performance, choose a control area of one cylinder. This can be accomplished if you allocate space in cylinders, or if you specify a number of kilobytes, megabytes, records, or tracks which are one cylinder or larger.

**Restriction:** If you add entries to a catalog in ascending sequence, it may result in control interval splits. For more information see *z/OS DFSMS: Using Data Sets*.

# Estimating Catalog Size

Because an ICF catalog uses variable-length, spanned records, it is not possible to precisely calculate the amount of space that a catalog requires. However, since secondary extents do not cause problems for catalogs, it is not necessary to be precise in making a size estimate. The following information serves only as an approximation for your catalog space requirements.

## Estimating Space Requirements for a Tape Volume Catalog

A volume catalog (VOLCAT) is an ICF catalog that contains only tape library and tape volume entries. A *general* VOLCAT contains all tape library entries and any tape volume entries that do not point to a specific VOLCAT. A *specific* VOLCAT cannot contain tape library entries. It contains a specific group of tape volume entries based on the tape volume serial numbers (tape volsers).

Table 4 lists the tape library and tape volume entries and approximates the number of bytes needed to maintain information in the tape volume catalog. The values do not necessarily represent the length of any one record.

*Table 4. Estimated Space Needed by the Tape Volume Catalog*

| Record | Number of Bytes |
|---|---|
| tape library entry | 320 |
| tape volume entry | 275 |

**Restriction:** A specific VOLCAT cannot contain tape library entries. It contains a specific group of tape volume entries based on the tape volume serial numbers (tape volsers).

Use the following steps to estimate the space allocation for a tape volume catalog:

1. Estimate the number of tape library entries and tape volume entries to be cataloged in the VOLCAT. Use only tape volume entries for a specific VOLCAT because it cannot contain tape library entries.
2. Using these figures, determine the total space requirement, in bytes. This figure is the minimum amount of space that the VOLCAT requires.
3. Divide the total number of bytes by 1024 to determine the number of kilobytes, or by 1048576 to determine the number of megabytes. Round the result up to the nearest whole integer, and specify KILOBYTES or MEGABYTES as appropriate.

## Estimating Space Requirements for the BCS

The amount of space a BCS requires depends on the type and number of data sets and objects cataloged in it. The amount of space in the BCS used by each type of data set or object can vary, depending on the:
- Length of the data set or component names
- Number of volumes per data set
- Number of relationships between components
- Number of alternate indexes for a data set
- Number of paths
- Presence of security information
- Presence of Storage Management Subsystem information.

Table 5 on page 44 lists the various types of data sets and objects, and approximates the number of bytes needed in the BCS to maintain information about a data set or object of that type. The values do not necessarily represent the length of any one record.

Data sets used by NFS or DFM will have 455 bytes added to their published size to accommodate new cell types.

*Table 5. Estimated Space Needed for Each Type of Data Set or Object*

| Data Set or Object Type | Number of Bytes |
|---|---|
| generation data group | 350 |
| generation data set | 200 |
| alias entry (see note) | 150 |
| non-VSAM data set or OAM object collection | 200 |
| user catalog connector (see note) | 200 |
| VSAM entry-sequenced, linear, or relative record data sets | 400 |
| VSAM key-sequenced data set | 650 |
| alternate index | 400 |
| path | 190 |

**Note:** To simplify the calculation for a user catalog connector record and to account for all of the catalog's aliases, count each catalog connector as requiring 32400 bytes.

Use the following steps to estimate the space allocation for a BCS:

1. Estimate the number of each type of data set or object which will be cataloged in the BCS. Using these figures, determine the total space requirement, in bytes, for the BCS. This figure is the minimum amount of space that the BCS requires.
2. Increase this quantity by the amount of free space required, and add additional space to allow for growth and any inaccuracies in the calculation. For example, if you define free space as 20% of each control interval and area, multiply by 1.7.
3. Divide the total number of bytes by 1024 to determine the number of kilobytes, or by 1048576 to determine the number of megabytes. Round the result up to the nearest whole integer, and specify KILOBYTES or MEGABYTES as appropriate.
4. Choose an appropriate secondary allocation. It is best if the secondary allocation is larger than the equivalent of one cylinder, so that the control area is defined as one cylinder.

The number of kilobytes or megabytes in a control area of one cylinder for different IBM DASD devices can be determined by multiplying the number of tracks per cylinder by the track capacity, and dividing by 1024 (kilobytes) or 1048576 (megabytes).

If you want to allocate space in tracks or cylinders, you must perform some additional calculations:

1. Divide the total number of bytes (including free space) by the data control interval size. Round up to the nearest whole integer. This is the number of required data control intervals.

2. Determine the number of control areas required by dividing the number of required control intervals by the number of control intervals which fit into your selected control area. Round up to the nearest integer.

3. Specify TRACKS or CYLINDERS as appropriate. If you specify TRACKS, space can be allocated in cylinders if the number of tracks is more than one cylinder.

## Estimating Space Requirements for the VVDS

The VVDS contains VSAM volume records (VVRs) which hold information about VSAM data sets residing on the volume. The VVDS also contains non-VSAM volume records (NVRs) for SMS-managed non-VSAM data sets on the volume. If an SMS-managed non-VSAM data set spans volumes, only the first volume contains an NVR for that data set.

The system automatically defines a VVDS with 10 tracks primary and 10 tracks secondary space, unless you explicitly define it.

If you want to explicitly define a VVDS, for example, as part of volume initialization, it is appropriate to use TRACKS(10 10) for the space allocation. However, you can estimate the space needed by estimating the number and type of data sets that will reside on the volume. Then, use the information in Table 6 to estimate the total amount of space needed for the VVDS.

Data sets used by NFS or DFM will have 500 bytes added to their published size to accommodate new cell types

*Table 6. Estimated Space Needed by the VVDS*

| Data Set Type | SMS-managed Volume | Non-SMS-managed Volume |
|---|---|---|
| VSAM key-sequenced data set or alternate index | 530 | 480 |
| VSAM entry-sequenced or relative record data set | 370 | 320 |
| VSAM linear data set | 340 | 290 |
| non-VSAM data set | 100 | 0 |

**Note:** These numbers assume 3 qualifier data set and catalog names of 8 characters per qualifier. Key-sequenced data sets and alternate indexes are assumed to have 64 character keys. SMS class names are assumed to be eight characters.

To estimate the total amount of space needed:

1. Add the space required by each data set.

2. Add 8 kilobytes for use by the VVDS.

3. Multiply the result by 1.2 to leave room for errors in the calculation.

4. Divide the result by 1024 to determine the number of kilobytes, and round up to the nearest integer.

Specify the result in the KILOBYTES parameter of the DEFINE CLUSTER command. Choose an appropriate secondary space allocation, usually at least half of the primary allocation.

If you want to allocate space in tracks, you must divide the total number of bytes by 4096 (the control interval for a VVDS). Then, divide the result by the number of 4096 blocks per track for the device.

# Choosing Options to Adjust Catalog Performance

You can enhance catalog performance by carefully selecting VSAM attributes when you define the catalog. Some catalog characteristics cannot be altered once the catalog is defined, so you should pay careful attention when selecting catalog characteristics.

Because the BCS of a catalog is a VSAM key-sequenced data set, you can enhance BCS performance by using the same considerations that you would use for similar key-sequenced data sets. For more information on performance considerations for VSAM data sets, see *z/OS DFSMS: Using Data Sets*.

The following sections address VSAM attributes for performance and recommended selections as they apply to catalogs. All parameters discussed are used on the DEFINE USERCATALOG command.

## Specifying the Number of Concurrent Requests

It is possible to specify the number of concurrent read requests for a BCS with the STRNO attribute. You can have from 2 to 255 concurrent read requests. Only one write request is allowed at a given time.

Initially define the catalog with **STRNO(3)**. You can monitor the adequacy of this value by using the Resource Measurement Facility (RMF) to watch for enqueues on the resource **SYSZRPLW.***catname***, and I/O contention on catalog volumes. SYSZRPLW** is the major name of the resource, and *catname* (the name of the catalog) is the minor name. An indication of enqueue contention is given by the I/O service times, not necessarily the number and frequency of enqueues on this resource.

You can change the STRNO attribute with the access method services ALTER command. This will take effect following a close and subsequent reopen of the catalog. You can close the catalog using the MODIFY CATALOG,CLOSE console command. The next request to the catalog will cause the catalog to be reopened.

If a catalog request is for update, an ENQ with exclusive control is issued on the BCS itself. However, most requests to the BCS are read requests, and those result in shared enqueues on the BCS.

Because requests for the VVDS are provided for dynamically, the number of concurrent requests is variable. There can be multiple updates to the VVDS concurrently, if the updates do not change the length of the VVDS records, and if the updates occur in different control intervals. An update to a record gets an exclusive enqueue on a control interval, and if the update changes the length of the VVDS record, the update request gets an exclusive enqueue on the VVDS itself.

## Other Catalog Performance Options

All the options discussed in this section are alterable, except for RECORDSIZE. If you decide your initial selection is inappropriate, you can use the access method services ALTER command to change it.

### Buffers

You can specify buffer size on three different parameters:

#### BUFFERSPACE

The required buffer space for your catalog is determined by VSAM. You only need to specify this if you want extra buffer space for your catalog.

In general, allow VSAM to determine buffer space. The number of buffers specified applies to each processor.

**BUFND**

Specifies the number of buffers for transmitting data between virtual and auxiliary storage. The default, STRNO+1, is usually adequate. When a catalog contains large GDGs or other spanned records, the number should be increased accordingly to a minimum of MAXLRECL/CISIZE+STRNO.

**BUFNI**

Specifies the number of buffers for transmitting index entries between virtual and auxiliary storage. The default, STRNO+2, is adequate for 3 levels of index in the catalog. If the catalog index exceeds 3 levels of index, the minimum BUFNI equals the number of levels of index - 1 + STRNO.

**FREESPACE**

The free space allows catalog updates without an excessive number of control interval and control area splits.

The following parameter affects catalog performance but can only be specified when the catalog is initially defined.

**RECORDSIZE**

Allow this parameter to default. The maximum value should always be the default if large records could be generated. Large catalog records are generated for GDGs that have many GDSs. Master catalogs with user catalog connector records that contain a large number of aliases also have large catalog records.

# Defining a Basic Catalog Structure

Before you define a catalog, determine which attributes to assign to the catalog. "Sharing Catalogs Among Systems" on page 11 discusses catalog sharing and share options. "Choosing Options to Adjust Catalog Performance" on page 46 discusses performance attributes. Carefully consider these sections before you define a new catalog; some of the options you can specify when defining a catalog cannot be altered.

For a discussion of attributes which can be altered for an existing catalog, see "Altering Catalog Attributes" on page 63.

Use the access method services DEFINE USERCATALOG ICFCATALOG command to define the basic catalog structure of an ICF catalog. Use the access method services DEFINE USERCATALOG VOLCATALOG to define a catalog that only contains tape library and tape volume entries. See the z/OS DFSMS Access Method Services for more detail.

If the BCS is being defined on a volume which contains a VVDS, then the existing VVDS is cataloged in the newly created BCS. If there is no VVDS on the volume, the DEFINE USERCATALOG ICFCATALOG job allocates one for you, using the default attributes for the VVDS (see "Defining a VVDS (VSAM Volume Data Set)" on page 51 for information on defining a VVDS). If the job tries to create a VVDS and cannot (for example, if there is not enough space on the volume), the job terminates before the BCS is defined.

You must specify ICFCATALOG on the DEFINE USERCATALOG command. Otherwise, you will create a VSAM catalog, the default.

# Define an ICF Catalog

The following example defines a catalog with the attributes recommended in this publication. The catalog defined, SYS1.ICFCAT.TEST, is placed on volume SYS305, and is allocated with 15 megabytes primary and secondary space.

```
//DEFCAT   JOB ...
//DEFCAT   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DEFINE USERCATALOG           -
        ( NAME(SYS1.ICFCAT.TEST) -
          MEGABYTES(15 15)       -
          VOLUME(SYS305)         -
          ICFCATALOG             -
          FREESPACE(10 10)       -
          STRNO(3)               -
        DATA( CONTROLINTERVALSIZE(4096) -
              BUFND(4)  )        -
        INDEX( BUFNI(4)  )
 /*
```

Catalog definition can be simplified by creating a data class for catalogs. Then, you can define new catalogs using the data class, and the appropriate attributes are assigned according to the data class definition.

# Define an ICF Tape Volume Catalog - General

A general VOLCAT is the default tape volume catalog. A general VOLCAT contains all tape library entries and any tape volume entries that do not point to a specific VOLCAT. Each system can have access to only one general VOLCAT. The general VOLCAT must be defined prior to bringing the tape libraries online. All ICF General Tape Volume Catalogs must be defined in the Master Catalog of the processing system. All systems connected to the IBM 3495 Tape Library Dataserver must use the same high level qualifier for their tape volume catalogs. If different versions exist, different systems can be accessing different sets of tape volume catalogs. See "Defining Names for a Tape Volume Catalog" on page 50 for more detail. The procedure for specifying the tape volume catalog high level qualifier is shown in "Bypassing SYSCATxx with LOADxx" on page 33.

This example defines an SMS-managed tape volume catalog named SYS1.VOLCAT.VGENERAL.

```
//DEFVCAT    JOB     ...
//STEP1      EXEC    PGM=IDCAMS
//SYSPRINT   DD      SYSOUT=A
//SYSIN      DD      *
     DEFINE USERCATALOG -
           (NAME(SYS1.VOLCAT.VGENERAL) -
            VOLCATALOG -
            VOLUME(338001) -
            CYLINDERS(1 1))
  /*
```

# Define an ICF Tape Volume Catalog - Specific

A specific VOLCAT is a tape volume catalog that contains a specific group of tape volume entries based on the tape volume serial numbers (tape volsers). A specific VOLCAT cannot contain tape library entries. See "Defining Names for a Tape

Volume Catalog" on page 50 for more detail concerning restrictions when different systems are connected to the Tape Library Dataserver.

This example defines an SMS-managed tape volume catalog named SYS1.VOLCAT.VT. This tape volume catalog would contain all tape volume entries that have a label beginning with the character ″T″. For example, one entry could be TAPE01.

```
//DEFVCAT    JOB     ...
//STEP1      EXEC    PGM=IDCAMS
//SYSPRINT   DD      SYSOUT=A
//SYSIN      DD      *
    DEFINE USERCATALOG -
          (NAME(SYS1.VOLCAT.VT) -
          VOLCATALOG -
          VOLUME(338001) -
          CYLINDERS(1 1))
/*
```

# Defining Aliases for a User Catalog

To use a catalog, the system must be able to determine which data sets should be defined in that catalog. The simplest way to accomplish this is to define aliases for the catalog. "Choosing Aliases and an Alias Search Level" on page 21 discusses general considerations for choosing aliases. Before defining an alias, carefully consider the effect the new alias will have on old data sets. A poorly chosen alias could make some data sets inaccessible.

You can define aliases for the catalog in the same job in which you define the catalog by including DEFINE ALIAS commands after the DEFINE USERCATALOG command. You can use conditional operators to ensure the aliases are only defined if the catalog is successfully defined. After the catalog is defined, you can add new aliases or delete old aliases.

Catalog aliases are defined in the master catalog, which contains an entry for the user catalog. The number of aliases a catalog can have is limited by the maximum record size for the master catalog. If the master catalog is defined with the default record sizes, there is a practical maximum of 3000 aliases per catalog, assuming the aliases are only for high-level qualifiers. If you use multilevel aliases, fewer aliases per catalog can be defined.

You cannot define an alias if a data set cataloged in the master catalog has the same high-level qualifier as the alias. The DEFINE ALIAS command fails with a "duplicate data set name" error. For example, if a catalog is named PAYROLL.TESTSYS.ICFCAT, you cannot define the alias PAYROLL for any catalog.

# Example of Defining Aliases for a User Catalog

The following job defines two aliases for SYS1.ICFCAT.TEST, USER01 and PROJECTA:

```
//DEFALIAS JOB ...
//ALIAS    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
      DEFINE ALIAS -
            (NAME(USER01)      -
            RELATE(SYS1.ICFCAT.TEST))

      DEFINE ALIAS -
            (NAME(PROJECTA)    -
            RELATE(SYS1.ICFCAT.TEST))
 /*
```

The NAME parameter identifies the alias, and the RELATE parameter identifies the catalog for which the alias is being defined.

# Defining Names for a Tape Volume Catalog

Use the access method services DEFINE USERCATALOG VOLCATALOG to define a catalog that only contains tape library and tape volume entries. See the z/OS DFSMS Access Method Services for more detail. Table 7 describes the naming conventions for a tape volume catalog.

*Table 7. Naming Conventions for a Tape Volume Catalog*

| Name | Type |
|---|---|
| xxxxxxxx.VOLCAT.VGENERAL | general tape volume catalog |
| xxxxxxxx.VOLCAT.Vx | specific tape volume catalog |

The high level qualifier of the tape volume catalog, xxxxxxxx, is 1 to 8 characters. It is specified by the LOAD*xx* member in SYS1.PARMLIB. See "Bypassing SYSCATxx with LOAD*xx*" on page 33 for a description of the format. When the system is initialized with SYSP=*xx*, the tape volume catalog high level qualifier specified in the LOAD*xx* member is activated. After initialization, you should use the MODIFY CATALOG,REPORT command to verify that the proper tape volume catalog high level qualifier is selected. For an example of the MODIFY CATALOG,REPORT output showing the high level qualifier, see "Monitoring the Catalog Address Space" on page 107.

All systems connected to the Tape Library Dataserver must use the *same* high level qualifier for their tape volume catalogs. All systems must be initialized with a LOAD*xx* member containing the same tape volume catalog high level qualifier. If different versions exist, different systems can be accessing different sets of tape volume catalogs.

Another problem can occur when one system does not specify the high level qualifier. In this case, the tape volume catalog high level qualifier defaults to SYS1. A second system is initialized with a LOAD*xx* member specifying something other than SYS1 as the tape volume catalog high level qualifier. Table 8 on page 51 describes the types of errors that can occur when two different systems, using the Tape Library Dataserver, use different high level qualifiers for tape volume catalogs.

*Table 8. Errors When Using Different Tape Volume Qualifiers*

| Error | Description |
|---|---|
| not found | the target tape volume catalog does not exist |
| synchronization errors | systems are making updates to different versions of tape volume catalogs and are out of synchronization with the real inventory |

To guarantee accuracy, all systems connected to the Tape Library Dataserver should use the *same* high level qualifier for their tape volume catalogs.

# Defining a VVDS (VSAM Volume Data Set)

A VVDS can be defined either:

- Explicitly, using DEFINE CLUSTER; or
- Implicitly, when the first VSAM or SMS-managed data set is defined on the volume.

A VVDS is defined with the name SYS1.VVDS.V*volser*, where *volser* is the volume serial number of the volume containing the VVDS. SYS1.VVDS.V*volser* does not have to be cataloged in the master catalog.

An explicitly defined VVDS is not related to any BCS until a data set or VSAM object is defined on the volume. As data sets are allocated on the VVDS volume, each BCS with VSAM or SMS-managed data sets residing on that volume is related to the VVDS.

An explicit definition of a VVDS does not update any BCS and, therefore, can be performed before the first BCS in the installation is defined.

Explicitly defining a VVDS is usually appropriate when you are initializing a new volume. If you are not running SMS, and a volume already contains some non-VSAM data sets, it is appropriate to allow the VVDS to be defined implicitly with the default space allocation of TRACKS(10 10).

If you are explicitly defining a VVDS, see "Estimating Space Requirements for the VVDS" on page 45 for an explanation of how to estimate the size of the VVDS.

# Example of Defining a VVDS

The following job defines and allocates a VVDS on volume SER003 with 10 tracks of space:

```
//DEFVVDS JOB ...
//DEFINE  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
    DEFINE CLUSTER -
           (NAME(SYS1.VVDS.VSER003) -
           TRACKS(10 10) -
           VOLUMES(SER003) -
           NONINDEXED)
/*
```

# Using One Catalog As a Model for Another Catalog

When you define a BCS or VVDS, you can use an existing BCS or VVDS as a model for the new one. The attributes of the existing data set are copied to the newly defined data set, unless you explicitly specify a different value for an attribute. You can override any of a model's attributes.

If you do not want to change or add any attributes, you need only supply the entry name of the object being defined and the MODEL parameter. When you define a BCS, you must also specify the volume and space information for the BCS.

When you use the MODEL parameter, ensure that the job is not terminated due to allocation problems when you explicitly do any of the following:
- Specify a different type of device with the VOLUMES parameter.
- Change the size of records, buffer space, or control intervals with the RECORDSIZE, BUFFERSPACE, or CONTROLINTERVALSIZE parameters.
- Change the unit of allocation with the CYLINDERS, TRACKS, KILOBYTES, MEGABYTES, or RECORDS parameters.

When you explicitly specify any of the above parameters for the BCS or VVDS to be defined, you might need to make corresponding changes to other related parameters.

If MODEL is specified as a parameter of USERCATALOG, the following steps occur:
1. The attributes of the model are copied for the BCS being defined.
2. Any attributes explicitly specified as parameters in the DEFINE command override those of the model.

If MODEL is specified as a parameter of CLUSTER (at the cluster level) but is not specified as a subparameter of the DATA or INDEX parameter, the following steps occur:
1. The attributes of the model are copied for the VVDS being defined.
2. Any attributes explicitly specified as parameters of CLUSTER override those of the model for the VVDS.
3. The attributes of the model's data and index components are copied for the VVDS's data and index components.
4. Attributes explicitly specified as parameters of CLUSTER are reproduced to the data and index components, overriding those of the model.
5. Attributes explicitly specified with subparameters of the DATA or INDEX parameters override the previous two steps.

If MODEL is specified both as a subparameter of DATA or INDEX and as a parameter of CLUSTER, the following steps occur:
1. The attributes of the CLUSTER model are copied for the cluster entry of the defined VVDS.
2. Any attributes explicitly specified as parameters of CLUSTER override those of the model for the VVDS.
3. Attributes explicitly specified as parameters of CLUSTER are reproduced to the VVDS's data and index components.
4. Attributes of the model specified with the MODEL subparameter of the DATA or INDEX parameters are copied, overriding the previous step.

5. Attributes explicitly specified with the subparameters of the DATA or INDEX parameters are copied, overriding the previous two steps.

## Example of Using a Model to Define a BCS

In this example, the catalog SYS1.ICFCAT.NEWCAT is defined using SYS1.ICFCAT.MODEL as a model. The MODEL parameter specifies the name of the model, and the catalog which contains the entries for the model. Because catalogs contain their own entries, you must specify the name of the catalog twice in the MODEL parameter.

```
//DEFCAT2  JOB  ...
//MODEL    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
    DEFINE USERCATALOG -
           (NAME(SYS1.ICFCAT.NEWCAT) -
            ICFCATALOG -
            MODEL(SYS1.ICFCAT.MODEL SYS1.ICFCAT.MODEL) -
            VOLUME(VSER03) -
            CYLINDERS(15 5))
 /*
```

# Chapter 4. Maintaining Catalogs

A major task of managing catalogs is maintaining an existing configuration of catalogs. You might need to change the size or location of a catalog to improve performance or maintain data security, delete obsolete catalogs, or remove volumes from a system. This chapter discusses some of the major tasks of catalog maintenance.

## Retrieving Information From a Catalog or VTOC

Information about a catalog or data set can be obtained using:
- Access method services (IDCAMS)
- ISMF
- IEHLIST
- The DFSMS attribute call service IGWASMS
- The TSO service routine IKJEHCIR
- The macros CAMLST and SHOWCAT
- The Catalog Search Interface (CSI) option.

## Listing the Contents of a Catalog

You can list catalog records using the access method services LISTCAT command, or the ISMF line operator CATLIST. CATLIST produces the same output as LISTCAT, but places the output in a data set that can be browsed.

You can use the LISTCAT output to monitor VSAM data sets. The statistics and attributes listed can be used to help determine if you should reorganize, recreate, or otherwise alter a VSAM data set to improve performance or avoid problems.

For example, you can use the values for High Used RBA and High Allocated RBA to help avoid out-of-space conditions for a data set or catalog. If the High Used RBA is less than the High Allocated RBA, then at least one control area split can occur without adding another secondary extent to the data set. Of course, you do not need to be concerned about secondary extents unless the volume is full or the data set already has a large number of them. If a data set has a large number of secondary extents, you might want to recreate the data set in a single extent.

Most information concerning non-VSAM data sets is maintained in the VTOC.

The statistical information contained in the self-describing entries for a BCS is not correct. Catalog management does not maintain information about the statistical attributes of a BCS.

The cluster entry name for a BCS is 44 bytes of zeros. The name of the data component is the name you gave the catalog. The name of the index component is generated according to the regular VSAM rules. The rules for VSAM generated names are described in *z/OS DFSMS: Using Data Sets*.

### Using LISTCAT in Examples

A catalog's self-describing entries are contained in the catalog itself. A catalog's connector record, which associates the catalog to its aliases is contained in the master catalog.

When you are listing the self-describing entries for a catalog, specify the name of the catalog in the CATALOG parameter. For example, the following step can be used to list the self-describing entry for SYS1.ICFCAT.VSYS303:

```
//LSTSDENT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   LISTCAT ALL ENTRIES(SYS1.ICFCAT.VSYS303) -
      CATALOG(SYS1.ICFCAT.VSYS303)
/*
```

You can use LISTCAT to determine which VVDSs are connected to a BCS. You can use this information to determine which VVDSs to compare to a BCS when you use the DIAGNOSE command. For example, the following step lists the VVDSs connected to SYS1.ICFCAT.VSYS303:

```
//LSTVVDS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   LISTCAT LEVEL(SYS1.VVDS) CATALOG(SYS1.ICFCAT.VSYS303)
/*
```

You can use LISTCAT to list the aliases associated with a catalog. Specify ALL with the catalog name in the ENTRIES parameter. The aliases are listed in the Associations group for the user catalog. If you specify a catalog in the CATALOG parameter, specify the master catalog. The following example lists the aliases associated with SYS1.ICFCAT.VSYS303:

```
//LSTALIAS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   LISTCAT ALL ENTRIES(SYS1.ICFCAT.VSYS303)
/*
```

You can use LISTCAT to display all fields associated with tape library and tape volume entries. Specify LIBRARYENTRIES to list tape library entries. Specify VOLUMEENTRIES to list tape volume entries. This example lists the tape library entry named ATLLIB1. Specify ALL to list all information associated with the tape library entry ATLLIB1.

```
//LISTCLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
   LISTCAT ENTRIES(ATLLIB1) -
      LIBRARYENTRIES -
      ALL
```

This example lists all the tape volume entries whose names begin with the letters 'VA' in the tape library named ATLLIB1.

```
//LISTCLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
   LISTCAT ENTRIES(VA*) -
      VOLUMEENTRIES -
      LIBRARY(ATLLIB1) -
      ALL
```

For more details on using LISTCAT, see *z/OS DFSMS Access Method Services*. For details on using CATLIST, see *z/OS DFSMS: Using the Interactive Storage Management Facility*.

## Printing a Catalog or VVDS

You can print the contents of a BCS or VVDS with the PRINT command, but the only circumstance where it might be useful is when you need to determine which catalogs are connected to a VVDS. This might be necessary to determine which BCSs to specify in a DIAGNOSE command, or when you are recovering a volume.

The names of the first 36 BCSs connected to a VVDS are in the first record of the VVDS. If you print this record using the DUMP format, you can read the names of the BCSs in the character format portion of the dump.

The following step can be used to print the first record of a VVDS. Since VVDSs are not normally found by catalog searches, use the INFILE parameter to specify a DD statement defining the VVDS.

```
//PRNTVVDS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//VVDS     DD DSN=SYS1.VVDS.VSPOOL1,DISP=SHR,
//            UNIT=SYSDA,VOL=SER=SPOOL1,AMP=AMORG
//SYSIN    DD *
     PRINT INFILE(VVDS) COUNT(1)
/*
```

A catalog or VVDS can also be printed using DFSMSdss™.

## Listing a Volume Table of Contents (VTOC)

The most convenient and flexible way to generate a data set list from a VTOC is to use the data set application of ISMF. With ISMF, you can generate a list of data sets based on many different filtering criteria, and execute commands against the data sets listed. You can also save the lists.

The VTOCLIST line operator can also be used to generate a VTOC listing for a data set in the IEHLIST format. For more information about using ISMF, see *z/OS DFSMS: Using the Interactive Storage Management Facility*.

If you want to use batch processing to get a VTOC listing, you can use the IEHLIST utility. The listing can be formatted or dumped in hexadecimal. For more information, see *z/OS DFSMSdfp Utilities*.

## Obtaining Information from an Application Program

An application program can access a catalog and retrieve information or perform other tasks. This can be done by using macros, or by using calls to system services or programs like access method services (IDCAMS).

The following is a list of macros or other callable services that can be used with ICF catalogs, and a brief description of their use:

*Table 9. Macros and System Services for Accessing Catalogs*

| Service | Description |
| --- | --- |
| CATALOG and CAMLST CAT | Macros used to catalog a non-VSAM data set. |
| CATALOG and CAMLST RECAT | Macros used to recatalog a non-VSAM data set. |
| CATALOG and CAMLST UNCAT | Macros used to uncatalog a non-VSAM data set, if it is not managed by the Storage Management Subsystem. |

*Table 9. Macros and System Services for Accessing Catalogs  (continued)*

| Service | Description |
|---|---|
| CATALOG SEARCH INTERFACE | Provides an alternative to LISTCAT that allows customizing of the output. You can display as little or as much information as you want. Also includes some information that LISTCAT does not provide. Three Assembler H sample programs and one REXX program are provided as part of CSI. See "Appendix B. Catalog Search Interface User's Guide" on page 141. For more information, see *IBM High Level Assembler/MVS & VM & VSE Programmer's Guide.* |
| IGWASMS | A DFSMS attribute call service used to identify the SMS classes for a data set, and determine whether the data set is a PDSE. For more information, see *z/OS DFSMSdfp Advanced Services.* |
| IKJEHCIR | A TSO programming service which can be used to retrieve a list of data sets with a matching high-level qualifier, and their associated data set types. It can also be used to identify the volume serial numbers and device types associated with a name, and find the next level qualifiers for a name. For more information, see *z/OS TSO/E Programming Services.* |
| LOCATE and CAMLST NAME | Macros used to retrieve information about a data set. For more information, see *z/OS DFSMSdfp Advanced Services.* |
| SHOWCAT | A macro used to retrieve information about a VSAM data set or generation data group. For more information, see *z/OS DFSMS Macro Instructions for Data Sets.* |

# Changing the Size or Contents of a Catalog

The main issues concerning the size or content of a catalog are availability and recoverability, not performance. Catalog performance is normally unaffected by the size of the catalog, or by the aliases defined for it.

When deciding whether a catalog is too big or small, or whether it has the wrong combination of entries (based on catalog aliases), consider how loosing that catalog would affect your installation with respect to how long it would take to recover the catalog.

Splitting, merging, or reorganizing catalogs can be time-consuming and disruptive to your users. Therefore, only perform these actions after carefully weighing the possible benefits against the lost availability of the catalog.

# Splitting Catalogs or Moving Catalog Entries

You can split a catalog into two catalogs or to move a group of catalog entries, if you determine that a catalog is either unacceptably large or that it contains too many entries for critical data sets.

If the catalog is unacceptably large (a catalog failure would leave too many entries inaccessible), then you can split the catalog into two catalogs. If the catalog is of an acceptable size but contains entries for too many critical data sets, then you can simply move entries from one catalog to another.

**Attention:**   Performing REPRO on a catalog while data sets are open in the source catalog might result in a loss of information if any of those data sets extend, or other catalog updates are made. The changes might not be copied to the target catalog, resulting in a mismatch between the information contained in the VVDS and the new target BCS. This might cause the data sets to be inaccessible or receive errors when they are used.

To split a catalog or move a group of entries, use the access method services REPRO MERGECAT command. The following steps should be followed to split a catalog or to move a group of entries:

1. Use ALTER LOCK to lock the catalog. If you are moving entries to an existing catalog, lock it as well.

2. If you are splitting a catalog, define a new catalog with DEFINE USERCATALOG LOCK.

3. Use LISTCAT to obtain a listing of the catalog aliases which you are moving to the new catalog. Use the OUTFILE parameter to define a data set to contain the output listing.

4. Use EXAMINE and DIAGNOSE to ensure that the catalogs are error-free. Fix any errors indicated.

5. Use REPRO MERGECAT to split the catalog or move the group of entries. When splitting a catalog, the OUTDATASET parameter specifies the catalog created in step 2. When moving a group of entries, the OUTDATASET parameter specifies the catalog which is to receive the entries. This step can take a long time to complete.

   Use the ENTRIES or LEVEL parameters to specify which catalog entries are to be removed from the source catalog and placed in the catalog specified in OUTDATASET.

   If this step fails for any reason, see "Recovering from a REPRO MERGECAT Failure" on page 60.

6. Use the listing created in step 3 to create a sequence of DELETE ALIAS and DEFINE ALIAS commands for each alias. These commands delete the alias to the original catalog, and redefine them as aliases for the catalog which now contains entries belonging to that alias name.

   The DELETE ALIAS/DEFINE ALIAS sequence must be run on each system that shares the changed catalogs.

7. Unlock both catalogs using ALTER UNLOCK.

## Merging Catalogs

You might find it beneficial to merge catalogs if you have many small or seldom-used catalogs. An excessive number of catalogs can complicate recovery procedures and waste resources such as CAS storage, tape mounts for backups, and system time performing backups.

Merging catalogs is accomplished in much the same way as splitting catalogs. The only difference between splitting catalogs and merging them is that in merging, you want all the entries in a catalog to be moved to a different catalog, so that you can delete the obsolete catalog.

The following steps should be followed to merge two ICF catalogs:

1. Use ALTER LOCK to lock both catalogs.

2. Use LISTCAT to list the aliases for the catalog you intend to delete after the merger. Use the OUTFILE parameter to define a data set to contain the output listing.

3. Use EXAMINE and DIAGNOSE to ensure that the catalogs are error-free. Fix any errors indicated.

4. Use REPRO MERGECAT *without* specifying the ENTRIES or LEVEL parameter. The OUTDATASET parameter specifies the catalog that you are keeping after the two catalogs are merged. This step can take a long time to complete.

5. Use the listing created in step 2 on page 59 to create a sequence of DELETE ALIAS and DEFINE ALIAS commands to delete the aliases of the obsolete catalog, and to redefine the aliases as aliases of the catalog you are keeping.

   If this step fails for any reason, see "Recovering from a REPRO MERGECAT Failure".

6. Use DELETE USERCATALOG to delete the obsolete catalog. Specify RECOVERY on the DELETE command.

   If your catalog is shared, run the EXPORT DISCONNECT command on each shared system to remove unwanted user catalog connector entries.

7. Use ALTER UNLOCK to unlock the remaining catalog.

You can also merge entries from one tape volume catalog to another using REPRO MERGECAT. REPRO retrieves tape library or tape volume entries and redefines them in a target tape volume catalog. In this case, VOLUMEENTRIES needs to be used to correctly filter the appropriate entries. The LEVEL parameter is not allowed when merging tape volume catalogs.

# Recovering from a REPRO MERGECAT Failure

Depending on the number of entries being processed, REPRO MERGECAT can require a lot of time to complete execution. If the REPRO MERGECAT job fails, for example, if the system goes down during execution, first determine what caused the job to fail.

As REPRO MERGECAT processes a catalog, it transfers entries from one catalog to another, deleting the entries from the source catalog. Thus, if REPRO MERGECAT has not completed, the target catalog contains the only valid entries for some of your data sets. For that reason, *do not* delete the target catalog simply because REPRO MERGECAT failed.

If the failure was caused by errors external to catalog management and access method services, simply rerun the REPRO MERGECAT job.

If the REPRO MERGECAT job cannot complete, use REPRO MERGECAT to *return* the entries from the target catalog which were moved there from the source catalog. This should return you to your starting point. Use the access method services DIAGNOSE command to help determine the catalog errors which must be corrected. Also, use EXAMINE to check the internal structure of the catalog.

Entries moved by REPRO MERGECAT are now cataloged in the new catalog, and they cannot be referenced from the old catalog. You should not use REPRO MERGECAT to make a backup copy of a catalog, or to create new catalog for another system. Using REPRO MERGECAT will make it impossible to refer to the data sets from the original catalog.

To help avoid REPRO MERGECAT failures, you should always use EXAMINE and DIAGNOSE before using REPRO MERGECAT, and fix any indicated errors. If REPRO MERGECAT cannot process a record because of an error in the record, it normally bypasses the record and issues a message. After using REPRO MERGECAT, carefully inspect all messages to determine if any entries were skipped. Then use DIAGNOSE on any skipped entries to determine the errors and the appropriate recovery procedures.

# Changing the Size of a BCS

You can change the size of a catalog if the catalog is much larger than necessary, or if it has grown into excessive secondary extents. Before changing the size of a catalog, consider merging small catalogs or splitting large catalogs.

When you change the size of a catalog, the catalog is also reorganized.

If you are decreasing the size of the catalog, you must be certain to define the new catalog with enough space to contain all the entries in the existing catalog. The catalog must also account for the free space defined for the catalog.

The following steps show how to change the size of a catalog. The catalog used in the example is called ICFCAT.USER.VSYS303.

1. Lock the catalog.

```
//LOCKCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   ALTER ICFCAT.USER.VSYS303 LOCK
/*
```

2. Export the BCS with the EXPORT command. The aliases of the catalog are saved with the exported copy, and can be used in later steps to redefine the aliases.

```
//EXPORT  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//EXPDD    DD DSN=CATBACK.ICFCAT.USER.VSYS303,DISP=OLD
//SYSIN    DD *
   EXPORT ICFCAT.USER.VSYS303 -
       OUTFILE(EXPDD) -
       TEMPORARY
/*
```

3. Delete the BCS with the RECOVERY option.

```
//DELCAT   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE ICFCAT.USER.VSYS303 -
       RECOVERY -
       USERCATALOG
/*
```

4. Redefine the BCS with the desired space and performance attributes.

```
//DFNEWCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   DEFINE USERCATALOG -
          (NAME(ICFCAT.USER.VSYS303) -
           VOLUME(SYS303) -
           MEGABYTES(15 5) -
           ICFCATALOG -
           FREESPACE(20 20) -
           STRNO(3)  -
           REPLICATE )  -
       DATA( CONTROLINTERVALSIZE(4096) -
             BUFND(4)  )  -
       INDEX( BUFNI(4)  )
/*
```

5. Import the BCS using the IMPORT command. Specify INTOEMPTY on the IMPORT command.

   Also, specify ALIAS, so that the aliases exported with the catalog are redefined.

You can also specify UNLOCK to unlock the catalog (UNLOCK is the default).

```
//IMPORT  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN   DD *
    IMPORT INDATASET(CATBACK.ICFCAT.USER.VSYS303) -
           OUTDATASET(ICFCAT.USER.VSYS303)       -
           ALIAS   -
           UNLOCK  -
           INTOEMPTY
 /*
```

6. If you did not unlock the catalog when you imported it, unlock it with the ALTER UNLOCK command.

# Recovering from a REPRO NOMERGECAT Failure

If the job fails, determine what caused the REPRO NOMERGECAT to fail. Because the source catalog is copied into an empty target catalog, the REPRO operation cannot be restarted if an error occurs.

Before you restart the copy operation, you must delete and redefine the target catalog. Make sure volumes, that contain objects to be copied, are restored.

After a successful REPRO of one catalog into an empty target catalog, the VVRs are changed to point to the target catalog. Subsequent processing occurs in the target catalog; entries can no longer be deleted from the source catalog.

# Changing the Size of a VVDS

A VVDS automatically reclaims space used by deleted records. It grows into secondary extents only when there is no space for a new record. Even if the VVDS grows into a few secondary extents, the performance and reliability of the VVDS are not affected.

If you still need to change the size of a VVDS, you must backup all data sets that have an entry in the VVDS, remove these data sets from the volume, delete the VVDS, redefine the VVDS, and finally restore the data sets from their backups to the volume.

**Restriction:** A VVDS is not subject to control interval splits.
The following steps are suggested for this process:

1. Obtain exclusive use of the volume by: stopping work, varying the volume offline to sharing systems, or quiescing sharing systems.
2. Remove all data sets from the volume that have an entry in the VVDS. All VSAM, SMS-managed data sets, and their catalogs should be removed using DFSMSdss (logical dump) or access method services (EXPORT, or REPRO followed by DELETE).

   With volumes that are not SMS-managed, you can use DFSMSdss to dump all VSAM data sets from the volume with one command. With SMS-managed volumes, use DFSMSdss to dump the entire volume.
3. Delete the VVDS specifying RECOVERY.
4. Define a new VVDS specifying the desired size. The default recommended size is trk(10,10).
5. Restore the data originally on the volume, using DFSMSdss™ or access method services (use the same utility used to dump the data sets).

You can use DIAGNOSE to determine any differences between the rebuilt VVDS and each BCS that had data sets on the volume. Because all existing data should have been recovered to the volume, there should be no differences.

# Renaming a Catalog

You cannot directly rename a catalog with the ALTER command. To rename a catalog, you must define a new catalog with the desired name and copy the old catalog into it. You can do this with the REPRO NOMERGECAT command.

REPRO NOMERGECAT is used to copy a catalog into an empty target catalog. Refer to *z/OS DFSMS Access Method Services* to determine if the REPRO NOMERGECAT option is appropriate for you.

All aliases to the old catalog must be redefined as aliases of the new catalog. Aliases are not automatically oriented to the new catalog.

The following steps show how to rename ICFCAT.USER.VSYS303 to ICFCAT.REPRO.VSYS303:

1. Before you rename a catalog, make sure it does not have structural and logical errors; to do this, run IDCAMS DIAGNOSE and EXAMINE jobs.
2. When you use NOMERGECAT, recovery is required in the event of a failure. Make sure you have a current back up of the catalog.
3. Lock the old catalog using ALTER LOCK.
4. List the aliases of the old catalog using LISTCAT. Specify the name of the old catalog in the ENTRIES parameter, and specify the ALL parameter. This lists all the aliases of the catalog in the ASSOCIATIONS group of the LISTCAT listing.

   Specify a data set using the OUTFILE parameter to contain the LISTCAT listing. This data set can be used in step 8.
5. Define the new catalog with the desired name, size, and other attributes. You can use the old catalog as a model.
6. Copy the original catalog into the new catalog using the REPRO NOMERGECAT command. Specify the old catalog in the INDATASET parameter, and the new catalog in the OUTDATASET parameter. Do not specify any other parameters.

   Refer to "Recovering from a REPRO NOMERGECAT Failure" on page 62.

   You can use REPRO with the MERGECAT option for this step. Refer to *z/OS DFSMS Access Method Services* to determine if the MERGECAT option is appropriate for you.
7. Delete the original catalog with the RECOVERY option. This also deletes the old aliases.
8. Use the listing of aliases created with LISTCAT to create a sequence of DEFINE ALIAS commands for each alias. Specify the name of the new catalog in the RELATE parameter.

# Altering Catalog Attributes

When you initially define a catalog, you choose its attributes according to your expectations of what the catalog requires. These requirements can change over time. However, only a portion of a catalog's attributes can be altered.

The following are the important attributes that can be altered:
• Buffer sizes (BUFFERSPACE, BUFND, BUFNI)
• FREESPACE

- MANAGEMENTCLASS
- SHAREOPTIONS
- STORAGECLASS
- STRNO
- WRITECHECK.

To alter these alterable attributes:

1. Use the access method services ALTER command to change the desired attribute. For complete information on the ALTER command, and for examples of altering catalog attributes, see *z/OS DFSMS Access Method Services.*

2. Close the catalog with MODIFY CATALOG,CLOSE so that the CAS control blocks for the catalog are refreshed.

The following attributes are unalterable:
- CONTROLINTERVALSIZE
- DATACLASS
- RECORDSIZE
- REPLICATE or NOREPLICATE for index records.

If you want to alter attributes which are unalterable:

1. Lock the catalog using ALTER LOCK.

2. Export the catalog with EXPORT.

3. Delete the catalog with DELETE RECOVERY.

4. Define a catalog with the same name on the same device with the desired attributes. Specify LOCK so that the new catalog cannot be used.

5. Import the catalog into the newly defined catalog with IMPORT INTOEMPTY. Allow the command to default to UNLOCK, so that the catalog is unlocked.

# Moving, Connecting, and Disconnecting Catalogs

You can move a catalog to another volume, even to a volume of a different device type, by using the EXPORT and IMPORT commands or by using DFSMSdss.

When you use IMPORT CONNECT for a tape volume catalog, you need to specify VOLCATALOG.

You can also move the catalog to another system by connecting it to the system and disconnecting it from the original system

If you connect a catalog to a sharing system, ensure that the share options for the catalog are (3 4); this is the default for the DEFINE USERCATALOG command. If the share options are (3 3), first use the ALTER command to change the share options to (3 4). The device on which the catalog resides must be defined as shared for each system.

# Moving a Catalog to a Different Volume

The procedure for moving a catalog to a different volume is essentially the same as that described in "Changing the Size of a BCS" on page 61.

The following steps show how to move a catalog. The catalog used in the example is called ICFCAT.USER.VSYS303.

1. Lock the catalog.

```
//LOCKCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   ALTER ICFCAT.USER.VSYS303 LOCK
/*
```

2. Export the BCS with the EXPORT command. The aliases of the catalog are saved with the exported copy, and can be used in later steps to redefine the aliases.

```
//EXPORT  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//EXPDD    DD DSN=CATBACK.ICFCAT.USER.VSYS303,DISP=OLD
//SYSIN    DD *
   EXPORT ICFCAT.USER.VSYS303 -
       OUTFILE(EXPDD) -
       TEMPORARY
/*
```

3. Delete the BCS with the RECOVERY option.

```
//DELCAT   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE ICFCAT.USER.VSYS303 -
       RECOVERY -
       USERCATALOG
/*
```

4. Redefine the catalog specifying the new volume. In the example, the new volume is SYSNEW. You may also change the space and performance attributes at this time.

```
//DFNEWCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   DEFINE USERCATALOG -
         (NAME(ICFCAT.USER.VSYS303) -
          VOLUME(SYSNEW) -
          MEGABYTES(15 5) -
          ICFCATALOG -
          FREESPACE(20 20) -
          STRNO(3)  -
          REPLICATE ) -
      DATA( CONTROLINTERVALSIZE(4096) -
          BUFND(4)  ) -
      INDEX( BUFNI(4)  )
 /*
```

5. Import the BCS using the IMPORT command. Specify INTOEMPTY on the IMPORT command.

   Also, specify ALIAS, so that the aliases exported with the catalog are redefined.

   You can also specify UNLOCK to unlock the catalog (UNLOCK is the default).

```
//IMPORT   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   IMPORT INDATASET(CATBACK.ICFCAT.USER.VSYS303) -
         OUTDATASET(ICFCAT.USER.VSYS303)        -
         OBJECTS((ICFCAT.USER.VSYS303   -
           VOLUMES(SYSNEW))        -
         ALIAS   -
         UNLOCK  -
         INTOEMPTY
 /*
```

6. If you did not unlock the catalog when you imported it, unlock it with the ALTER UNLOCK command.

Alternatively, if you do not want to do the DELETE, DEFINE, and IMPORT described in steps 3. through 5., you can import the catalog to the new volume using the IMPORT command with the OBJECTS parameter. Note that IMPORT deletes the old catalog on the previous device and moves the catalog to the specified different volume.

```
//NEWVOL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
    IMPORT INDATASET(CATBACK.ICFCAT.USER.VSYS303) -
           OUTDATASET(ICFCAT.USER.VSYS303) -
           OBJECTS((ICFCAT.USER.VSYS303 -
                   VOLUMES(SYSNEW)))   -
           ALIAS -
           LOCK
/*
```

In this example, the new volume is SYSNEW, and is specified in the OBJECTS parameter.

If the target device type has different device characteristics, the new catalog can have different data and index component control interval sizes. If you do not want the attributes of the catalog to change use the method described in steps 3. through 5.; that is, delete the catalog with DELETE RECOVERY, define a catalog with the desired attributes on the new volume, and import the original catalog into the newly defined catalog.

VSAM clusters with RACF discrete profiles use the catalog volume for RACF discrete profile checking. If a catalog is moved to a new volume, all VSAM clusters with a discrete profile that reside in that catalog *must* have their RACF discrete profiles updated with the new catalog volume.

### Updating Catalog Connector Records

A connector record is a record in the master catalog for a user catalog. For shared catalogs, this record requires updating if another system has moved the catalog.

If a connector record must be updated due to a change in device type or volume serial number for the catalog, use the IMPORT CONNECT ALIAS command. Specifying ALIAS preserves any aliases already defined for the catalog.

Use the OBJECTS parameter to indicate the changed device type and volume serial number.

A moving user catalogs example follows:

```
//IMPORTCN EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
  IMPORT CONNECT ALIAS -
     OBJECTS((ICFCAT.USER.VSYS303 VOLUMES(SYSNEW) -
        DEVT(3390))) -
     CAT(SYS1.MASTERB.ICFCAT)
/*
```

## Moving a Catalog to a Different System

You might need to move a catalog from one system to another. In this case, you are not necessarily changing the physical location of the catalog, but you are changing how the catalog can be accessed and used.

To move a catalog from one system to another system, the catalog must reside on a volume used by the receiving system. If the volume containing the catalog is already shared by the receiving system, you do not have to change the physical location of the catalog. If the volume is not shared, move the catalog to a volume used by the receiving system.

Once the catalog is on a volume that the receiving system can access, use the IMPORT CONNECT command to connect the catalog to the system's master catalog. If you do not want the original system to use the catalog, use EXPORT DISCONNECT to break the connection between the original master catalog and the user catalog you are moving. If the catalog's share options are (3 4) and the device is defined as shared, the catalog can be shared by both systems.

It is not necessary to lock a catalog when using IMPORT CONNECT or EXPORT DISCONNECT. Jobs oriented to a catalog should end normally if the catalog is disconnected. However, if you are permanently moving a catalog to a different system, ensure that users remove references to the catalog from their jobs.

When you use IMPORT CONNECT, the aliases defined for the catalog on one system are not transferred or redefined on the receiving system, even if you specify ALIAS. If you are permanently moving a catalog to a different system, simply import an exported copy of the catalog (without specifying CONNECT), naming the receiving system's master catalog in the CATALOG parameter. Specify ALIAS, so that the aliases are redefined.

Another way to preserve aliases is to use LISTCAT to obtain a list of the aliases. Then, after you have connected the catalog to the receiving system, use DEFINE ALIAS on the receiving system to define the aliases.

The following example shows how to connect a user catalog on SYSTEMA to SYSTEMB using IMPORT CONNECT, and then how to remove the catalog from SYSTEMA. Before removing the catalog from SYSTEMA, the aliases to the catalog are listed, so they can later be defined for the catalog on SYSTEMB. The LISTCAT output is directed to a data set which can later be edited to produce a series of DEFINE ALIAS commands to define the aliases on the receiving system. The alias associations are listed in the Associations group in the LISTCAT output.

```
//SYSMVCT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//LISTDATA DD DSN=LISTCAT.ALIAS.LISTING,DISP=OLD
//SYSIN    DD *
    LISTCAT ALL -
            ENTRIES(ICFCAT.USER.VSYS303) -
            OUTFILE(LISTDATA)

    IMPORT CONNECT -
            OBJECTS((ICFCAT.USER.VSYS303 -
               DEVICETYPE(3390) -
               VOLUMES(SYS303))) -
            CATALOG(SYSTEMB.MASTER.CATALOG)

    EXPORT ICFCAT.USER.VSYS303 -
            DISCONNECT -
            CATALOG(SYSTEMA.MASTER.CATALOG)
/*
```

# Establishing and Breaking Connections between BCSs and VVDSs

Occasionally, you might need to establish or break a connection between the master catalog and a user catalog, or between BCSs and VVDSs. For example, the DIAGNOSE command issues a message if a BCS and VVDS are connected, but there are no data sets on the VVDS's volume which are cataloged in the connected BCS. If this is the expected condition, breaking the connection between the VVDS and the BCS eliminates unwanted messages from DIAGNOSE.

See Figure 4 on page 23 for an illustration of how master and user catalogs, and VVDSs, are connected.

A BCS is connected to the master catalog of each system which shares the catalog. However, when the BCS is defined, it is only connected to the master catalog on the same system. To connect the catalog to other sharing systems, use the IMPORT CONNECT command. A catalog does not have to exist in order to connect it to a master catalog.

If you specify ALIAS on the IMPORT CONNECT command, any aliases which are already defined in the target master catalog for the catalog are maintained. Aliases which are not already defined in the target master catalog are not added, however. ALIAS can be specified even if no aliases are defined for the catalog.

Alternatively, you might need to disconnect a catalog from a system. In this case, use EXPORT DISCONNECT. This command removes the catalog's entry from the master catalog, and deletes all associated aliases. You do not have to lock a catalog before disconnecting it. Any jobs oriented to the catalog should end normally.

The catalog does not have to exist in order to disconnect it. For instance, if another system has deleted the catalog, any sharing system can disconnect that catalog.

You cannot use DELETE NOSCRATCH to delete a catalog's entry in the master catalog.

The system connects VVDSs with BCSs as the need arises. However, if you want to explicitly establish a connection between a VVDS and a BCS, you can use DEFINE RECATALOG, specifying the VVDS as the entry name, and specifying the BCS in the CATALOG parameter.

To break the connection between a BCS and a VVDS, use DELETE NOSCRATCH, specifying the VVDS as the entry name, and the BCS in the CATALOG parameter. If the VVDS is available, it is checked to determine if the BCS in the CATALOG parameter has any data sets on the volume. If data sets are found on the volume, the DELETE command will fail. If the VVDS is not available, no check is performed.

# Deleting Catalogs and Catalog Entries

Use the access method services DELETE command to delete catalogs, cataloged data sets, objects, tape library entries, and tape volume entries. With this command, you can delete anything you can define or create with access method services.

The access method services ALTER, CREATE, and DELETE commands should only be used to *recover* from tape volume catalog (VOLCAT) errors. Since access method services cannot change the library manager inventory in an automated tape

library, ISMF should be used for normal tape library ALTER, CREATE, and DELETE functions. DELETE can also be used for deleting erroneous records from BCSs, VVDSs, and VTOCs.

This section focuses on deleting catalogs and other system or sensitive data. For information on using DELETE to delete the entries for a data set (not the data set itself) from a BCS, VVDS, or VTOC, see "Updating the Catalog After Recovery" on page 90.

Before using the DELETE command, review the command description in *z/OS DFSMS Access Method Services*.

## Deleting Catalogs for Recovery

When you are recovering a catalog, you can temporarily delete the catalog without deleting any data sets cataloged in it. Normally, it is not necessary to delete a catalog before recovering it. If you want to change the size or unalterable attributes of a catalog, however, delete the catalog for recovery, define the catalog with the desired size and attributes, and then recover the catalog.

To delete a catalog for recovery, use the DELETE RECOVERY command. When used on a BCS, DELETE RECOVERY:
* Deletes the connector record for the BCS from the master catalog
* Deletes all aliases for the catalog from the master catalog
* Deletes the VVDS records which describe the BCS
* Deletes the VTOC DSCBs for the BCS
* Does not delete or alter any data set cataloged in the BCS.

You can also delete a VVDS for recovery using DELETE RECOVERY. This is normally only done when you are rebuilding a VVDS. When used on a VVDS, DELETE RECOVERY:
* Deletes the entry for the VVDS from the specified BCS
* Deletes the VTOC DSCB for the VVDS
* Does not delete the BCS and VTOC entries for the data sets and VSAM objects which were reflected in the VVDS
* Does not delete the entries for the VVDS in any other BCSs.

Before deleting a VVDS, ensure that sharing systems have been quiesced or have varied the shared DASD offline.

You cannot delete a VVDS which contains entries for data sets cataloged in the master catalog. Since any VSAM data set or SMS-managed data set is inaccessible after the VVDS is deleted, delete any BCS on the volume before rebuilding the VVDS. The BCS can later be recovered from a backup copy. If you can, export the BCS before deleting the VVDS.

If a VVDS is not cataloged in any BCS, you must establish a connection with a BCS before deleting the VVDS. This can be done with DEFINE RECATALOG, specifying the VVDS and a catalog in the CATALOG parameter. It can also be done by defining a VSAM data set on the volume with the VVDS. You can then delete the VSAM data set before deleting the VVDS.

## Deleting a Catalog Permanently

In general, a catalog should be empty before you permanently delete it. However, you can delete the catalog and all VSAM data sets, SMS-managed non-VSAM data sets, or objects cataloged in it if you choose.

If the BCS is empty (it contains only records describing itself and the VVDS on its volume), simply delete it using DELETE USERCATALOG. This removes the entries for the BCS from the VTOC and VVDS, and removes the entry in the master catalog. If the BCS is shared with other systems, use EXPORT DISCONNECT on the sharing systems to finish deleting the catalog.

If the BCS no longer exists, but has a connector record in the master catalog, use EXPORT DISCONNECT to delete the connector record.

If the BCS is not empty, and you want to delete the VSAM and SMS-managed data sets cataloged in it as well, specify FORCE on the DELETE USERCATALOG command. When you use FORCE:

- The BCS and its aliases are deleted.
- All VSAM data sets and objects cataloged in the BCS are deleted.
- All SMS-managed non-VSAM data sets are deleted.
- All non-VSAM data sets not managed by SMS are uncataloged, but they are not deleted.
- No VVDSs are deleted.

When you delete a non-empty catalog using FORCE, VSAM data sets and objects cannot be erased, even if you specify ERASE on the DELETE command. You should delete any sensitive data individually before deleting the non-empty catalog.

## Deleting a VVDS Permanently

A VVDS must be connected to a BCS before it can be deleted. A VVDS must be empty (contain only self-describing entries) before it can be permanently deleted. You cannot permanently delete a VVDS on an SMS-managed volume unless the volume is empty of data sets, because all managed data sets must have VVDS entries.

If the VVDS is not connected to any BCS, use DEFINE RECATALOG to catalog the VVDS. Alternatively, define a VSAM cluster on the volume with the VVDS. This catalogs the VVDS in the BCS.

## Deleting Catalog Aliases

To simply delete an alias, use the DELETE ALIAS command, specifying the alias you are deleting.

To delete all the aliases for a catalog, use EXPORT DISCONNECT, to disconnect the catalog. The aliases are deleted when the catalog is disconnected. When you again connect the catalog (using IMPORT CONNECT), the aliases remain deleted.

## Removing All VSAM Data from a Volume

You can delete all VSAM data sets on a volume with the ALTER REMOVEVOLUMES command if, and only if, the volume is not managed by the Storage Management Subsystem. If the master catalog is specified as the entry name on the ALTER REMOVEVOLUMES command and the target volume is SMS-managed, the volume becomes unusable and you must recover the volume.

When the master catalog is specified, the ALTER REMOVEVOLUMES command deletes all VSAM data sets from the specified volume, including any BCS or VVDS on the volume. If a BCS resides on a volume, first export the BCS before using ALTER REMOVEVOLUMES. You can then recover the catalog from the exported copy.

There are three reasons you might want to use ALTER REMOVEVOLUMES to perform VSAM volume cleanup:

> To recover all and only VSAM data sets on a volume,
> To delete a BCS on a volume with no VVDS, or
> To delete a VVDS which you cannot connect to a BCS.

If you are trying to recover only selected VSAM data sets, use the DELETE command instead of ALTER REMOVEVOLUMES.

If you are removing the VSAM objects from a volume, use the DIAGNOSE command before and after ALTER REMOVEVOLUMES, to check for lost data or other problems. Use both the ICFCATALOG and VVDS parameters of DIAGNOSE.

Since ALTER REMOVEVOLUMES deletes VVDSs, any connections between a VVDS and BCSs on other volumes are broken. You can reconnect the VVDS to the BCSs using DEFINE RECATALOG. However, when a data set on the volume is accessed, the VVDS is recataloged in the appropriate BCS automatically. After the VVDS is removed, you can delete the VVDS's record from BCSs on other volumes using DELETE NOSCRATCH.

## VSAM Record-Level Sharing

If you rename, alter, move, recover or change the size of a basic catalog structure when using VSAM record-level sharing (RLS), you must use the AMS SHCDS CFREPAIR command to correct RLS information in the catalog. The catalog must be import connected on all systems to the master catalog before the SHCDS CFREPAIR command can be used. If you are using VSAM RLS support and decide to no longer use it, the AMS SHCDS CFRESET command is used to reset applicable RLS indicators in the catalog. See *z/OS DFSMS Access Method Services* for more information about the SHCDS command.

## Deleting Sensitive Data

Typically, when you delete a data set, only the catalog, VVDS, and VTOC information is removed. The information on the disk or tape which the data set occupied is unchanged; only the means of locating and accessing the information is actually deleted. Until that space is used again, the information could be read by a program which can find the data.

To protect sensitive information, you can erase the information when you delete it. Information is erased by overwriting it with binary zeros before the space is made available for other allocations.

You can control the erasure of data with the Resource Access Control Facility (RACF), the DEFINE command, or the DELETE command. Data is erased according to the following rules:

1. If the RACF generic or discrete profile specifies ERASE, the data is erased.
2. If ERASE is specified on the DELETE command, the data is erased, even if the RACF profile specifies NOERASE.

3. If NOERASE is specified on the DELETE command, the data is erased only if the RACF profile specifies ERASE. If NOERASE is desired, use a RACF command to change the attribute before using the DELETE command.

4. If a VSAM cluster or alternate index is deleted, and it was defined with ERASE specified, the data is erased unless NOERASE is specified on the DELETE command.

For more information about the RACF ERASE attribute, see "RACF-Controlled ERASE Options" on page 78.

# Chapter 5. Protecting Catalogs

The protection of data includes:
- Data security—the safety of data from theft or intentional destruction
- Data integrity—the safety of data from accidental loss or destruction.

Data can be protected either indirectly, by preventing access to programs which can be used to modify data, or directly, by preventing access to the data itself. Catalogs and cataloged data sets can be protected in both ways.

To protect your catalogs and cataloged data, use the authorized program facility (APF) and the Resource Access Control Facility (RACF). VSAM passwords and USVRs are no longer used.

Because of its special use, you cannot export, or import a VVDS, nor can you alter its attributes by using the ALTER command.

# Authorized Program Facility Protection for Access Method Services

The authorized program facility (APF) limits the use of sensitive system services and resources to authorized system and user programs.

For information about using APF for program authorization, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

All access method services load modules are contained in SYS1.LINKLIB, and the root segment load module (IDCAMS) is link-edited with the SETCODE AC(1) attribute. These two characteristics ensure that access method services executes with APF authorization.

APF authorization is established at the job step task level. If a load request is satisfied from an unauthorized library during the execution of an APF authorized job step, the task is abnormally terminated. It is the installation's responsibility to ensure that a load request cannot be satisfied from an unauthorized library during access method services processing.

The following situations could cause the invalidation of APF authorization for access method services:
- An access method services module is loaded from an unauthorized library, or invoked by an unauthorized program.
- A user-security verification routine (USVR) is loaded from an unauthorized library during access method services processing.
- An exception installation or user exit routine is loaded from an unauthorized library during access method services processing.
- A user-supplied special graphics table is loaded from an unauthorized library during access method services processing.

Because APF authorization is established at the job step task level, access method services is not authorized if invoked by an unauthorized application or terminal monitor program.

Under the time sharing option (TSO), if the system does not have the TSO Command Package Program Product, you can authorize your terminal monitor program by link-editing it with the SETCODE AC(1) attribute. You must enter the

names of those access method services commands requiring APF authorization to execute under TSO in the authorized command list (AUTHCMD) in the SYS1.PARMLIB member IKJTSO*xx* or added to the CSECT IKJEGSCU. Please see *z/OS TSO/E Customization* for more information.

The restricted functions performed by access method services that cannot be requested in an unauthorized state are:
- DEFINE—when the RECATALOG parameter is specified
- DEFINE—when the define is for an alias of a UCAT
- DELETE—when the RECOVERY parameter is specified
- EXPORT—when the object to be exported is a BCS
- IMPORT—when the object to be imported is a BCS
- PRINT—when the object to be printed is an ICF catalog
- REPRO—when a BCS is copied or merged
- VERIFY—when a BCS is to be verified.
- SHCDS—all functions

# Resource Access Control Facility (RACF) Protection

You should use RACF to protect your data sets and catalogs. Only RACF and APF are used with SMS.

# RACF Authorization Checking

To open a catalog as a data set, you must have ALTER authority and APF authorization. When defining an SMS-managed data set, the system only checks to make sure the user has authority to the data set name and SMS classes and groups. The system selects the appropriate catalog, without checking the user's authority to the catalog. You can define a data set if you have ALTER or OPERATIONS authority to the applicable data set profile.

Deleting any type of RACF-protected entry from a RACF-protected catalog requires ALTER authorization to the catalog or to the data set profile protecting the entry being deleted. If a non-VSAM data set is SMS-managed, RACF does not check for DASDVOL authority. If a non-VSAM, non-SMS-managed data set is being scratched, DASDVOL authority is also checked.

Altering the passwords in a RACF-protected catalog entry requires ALTER authority to the entry being altered, or the OPERATIONS attribute. ALTER authority to the catalog itself is not sufficient for this operation.

For ALTER RENAME, the user is required to have the following two types of authority:
1. ALTER authority to either the data set or the catalog
2. ALTER authority to the new name (generic profile) or CREATE authority to the group.

Be sure that RACF profiles are correct after you use REPRO MERGECAT or CNVTCAT on a catalog that uses RACF profiles. If the target and source catalogs are on the same volume, the RACF profiles remain unchanged.

REPRO MERGECAT will preserve RACF discrete profiles when the target and source catalog are on different volumes. Profiles will be updated with the target volume, except when the protected data set is DFSMShsm™ migrated. Profiles for

DFSMShsm migrated data sets must be manually changed using RACF commands. Be sure to verify the integrity of discrete profiles after MERGECAT. You should use generic profiles to avoid this situation.

Non-VSAM tape data sets defined in an ICF catalog can be protected by:
- Controlling access to the tape volumes; or
- Controlling access to the individual data sets on the tape volumes.

Note that if you run RACF in ″warn″ mode, you may receive indications of access violations. Catalog processing uses two-step verification for many types of functions. The first test checks to see if the user has authority to the specific data set. If this request fails the security check, the system will attempt to verify if the user has the appropriate authority to the containing catalog. If this request succeeds, the access is granted. However, in warn mode a message will be produced for the first security check which failed, even though the user passes the stated security checks for the access. These messages can be ignored, as they will disappear when RACF is no longer running in 'warn' mode.

## Generic Profile-Checking Facility

RACF provides a generic profile-checking facility. With the always-call capability of ICF catalogs, you can consolidate the access authorization requirements of several similarly named and similarly used data sets under a single generic profile definition. A generic profile is used to protect one or more data sets that have identical access requirements. For example, you can build a generic profile named 'userid.*' to protect all data sets cataloged under the same high-level qualifier. For more information, see *z/OS Security Server RACF Command Language Reference*.

VSAM data sets that are protected by generic profiles are not RACF-indicated in the catalog. Therefore, RACF is always called for any access to data sets cataloged in ICF catalogs. If the data set is not protected by either a discrete profile or a generic profile, no protection is in effect. The ICF catalog does not have to be RACF-protected in order for its data sets to be RACF-protected.

For VSAM clusters cataloged in an ICF catalog, all the components of the VSAM cluster are protected by one profile (the profile that protects the cluster name). This profile can be discrete or generic. You do not need to create profiles that protect the index and data components of a cluster.

Data sets protected with discrete profiles are flagged as "RACF-indicated." If a data set protected by a discrete profile is moved to a system where RACF is not installed, no user is given authority to access the data set. However, if the data set is protected with a generic profile, it is not flagged as "RACF-indicated"; therefore, access authority is determined by normal VSAM password protection.

## Controlling Catalog Functions with RACF Profiles in the FACILITY Class

By defining and controlling access to profiles in the FACILITY class, you can control who can use certain catalog functions. Besides defining these profiles, you must activate the FACILITY class for these functions to be protected.

These profiles can be assigned to an owner. For example, the person responsible for managing catalogs can be assigned ownership of these profiles. A profile owner can then list, modify, or delete the profiles as needed.

The following RACF commands show how to define a FACILITY profile, authorize a user to perform the functions restricted by the profile, and activate the FACILITY class. The profile defined is IGG.CATLOCK, which is assigned to user CATADMIN, and user USER01 is authorized to use the profile.

```
RDEFINE  FACILITY  IGG.CATLOCK  UACC(NONE) OWNER(CATADMIN)

PERMIT  CLASS(FACILITY)  IGG.CATLOCK  ID(USER01) ACCESS(READ)

SETROPTS CLASSACT(FACILITY)
```

The RDEFINE command creates the profile and gives it a universal access authority (UACC) of NONE. Since READ authority to the profile allows a user to perform the protected function, you must use a UACC of NONE to limit the use of the protected function. The PERMIT command is used to authorize the appropriate users or groups to perform the protected function. If the FACILITY class is already active, the SETROPTS command is not necessary.

## Controlling Who Can Lock a Catalog (IGG.CATLOCK Profile)
The IGG.CATLOCK profile, in conjunction with normal security checking, controls who can lock a catalog and who can access a locked catalog.

If you have READ access to the IGG.CATLOCK profile and ALTER authority to the catalog, you can lock or unlock a catalog. If you have READ access to the IGG.CATLOCK profile, you can access and repair a locked catalog. If the IGG.CATLOCK profile is not defined, or the FACILITY class is not active, you cannot lock or unlock a catalog.

In previous releases, a user defined as privileged or trusted to RACF would automatically have read access to this facility class. However, you must now explicitly authorize those privileged or trusted users to this facility class if they need access to locked catalogs. There are products that are privileged (such as DFSMShsm) because they perform a variety of actions against data sets and catalogs. Without requiring explicit authorization to the facility class, these products could inadvertently access or change locked catalogs, causing damaged catalogs or unexpected results. If these products or components are to be used while accessing a locked catalog, they must be explicitly authorized to the facility class. It is recommended they be authorized only for the duration of the specific need.

To ensure the integrity of integrated catalogs, restrict authority to the IGG.CATLOCK profile to only the necessary people or system components. To define entries in a catalog, users only need UPDATE authority to the data set profile protecting the catalog. Therefore, you should consider specifying UACC(UPDATE) for the data set profiles protecting user catalogs. To delete entries in a catalog, users need either ALTER authority to the data set or ALTER authority to the catalog. We recommend that you only give users ALTER authority to their own data sets.

See "Locking a Catalog" on page 84 for an explanation of catalog locking, and "Recovering a BCS" on page 85 for an example of locking a catalog during recovery.

## Storage Administration (STGADMIN) Profiles in the FACILITY Class
To control the ability to perform functions associated with storage management, define profiles in the FACILITY class whose profile names begin with STGADMIN (storage administration). The STGADMIN.IGG facility classes are only intended for SMS data sets.

If defined, these profiles are checked before a user is allowed to perform the protected function. If these profiles are not defined, other RACF checking is still made to verify authority.

Some FACILITY profiles are not checked if the caller is using the system key or is running in supervisor state. These profiles are:

    STGADMIN.IGG.DEFDEL.UALIAS
    STGADMIN.IGG.DEFNVSAM.NOBCS
    STGADMIN.IGG.DEFNVSAM.NONVR
    STGADMIN.IGG.DELETE.NOSCRTCH
    STGADMIN.IGG.DELGDG.FORCE
    STGADMIN.IGG.DELGDG.RECOVERY
    STGADMIN.IGG.DELNVR.NOBCSCHK
    STGADMIN.IGG.DIRCAT
    STGADMIN.IGG.LIBRARY

Define the following classes to protect catalog functions. For a complete list of STGADMIN profiles, see *z/OS DFSMSdfp Storage Administration Reference*.

**STGADMIN.IDC.DIAGNOSE.CATALOG**
>       protects the ability to use the access method services DIAGNOSE command against catalogs.

**STGADMIN.IDC.DIAGNOSE.VVDS**
>       protects the ability to use the access method services DIAGNOSE command against a VVDS when a comparison is made to a BCS.

**STGADMIN.IDC.EXAMINE.DATASET**
>       protects the ability to use the access method services EXAMINE command on ICF catalogs.

**STGADMIN.IGG.ALTBCS**
>       protects the ability to alter BCS attributes.

**STGADMIN.IGG.ALTER.SMS**
>       protects the ability to alter the storage class or management class of an SMS-managed data set. If the profile is not built, the user must have authority to the class to alter it.

**STGADMIN.IGG.ALTER.UNCONVRT**
>       protects the ability to alter an SMS-managed VSAM data set to an unmanaged VSAM data set.

**STGADMIN.IGG.DEFDEL.UALIAS**
>       allows you to define or delete an alias related to a usercatalog without further authorization checking.

**STGADMIN.IGG.DEFNVSAM.NOBCS**
>       protects the ability to define a non-VSAM data set with no BCS entry. Only a VVDS record (an NVR) for an SMS-managed non-VSAM data set is created.

**STGADMIN.IGG.DEFNVSAM.NONVR**
>       protects the ability to define a non-VSAM data set with no VVDS entry (an NVR). Only a BCS entry for an SMS-managed non-VSAM data set is created.

**STGADMIN.IGG.DELETE.NOSCRTCH**
>       protects the ability to delete the BCS entry for an SMS-managed data set without deleting the data set itself (for example, using DELETE NOSCRATCH). This protects against functions which uncatalog data sets.

**STGADMIN.IGG.DELGDG.FORCE**
protects the ability to use DELETE FORCE on a generation data group which contains an SMS-managed generation data set. The DELETE GDG FORCE command deletes SMS generation data sets referenced by the generation data group. It also removes the generation data group entry in the catalog.

**STGADMIN.IGG.DELGDG.RECOVERY**
this command deletes the generation data group and uncatalogs the SMS generation data sets. When you use this command, the generation data group entry is deleted from the catalog and generation data sets remain unaffected in the VTOC, and if SMS managed, in the VVDS.

**STGADMIN.IGG.DELNVR.NOBCSCHK**
protects the ability to delete the VVDS entry (the NVR) for an SMS-managed non-VSAM data set and to bypass the catalog name and BCS entry checking. If there is a BCS entry or if the catalog name contained in the NVR does not match the catalog provided in the request, the function is denied unless the user has authority to this profile.

**STGADMIN.IGG.DIRCAT**
protects the ability to direct a catalog request to a specific catalog, bypassing the normal catalog search. A directed catalog request is one in which the catalog name is explicitly passed to catalog management in the CATALOG parameter of access method services commands.

In an SMS environment, all the catalog requests against SMS-managed data sets should be satisfied by the normal catalog search order. You must be authorized to this facility class in order to direct the catalog request to a specific catalog, unless you are using one of the following commands:
- LISTCAT
- DEFINE ALIAS of a usercatalog
- IMPORT CONNECT
- EXPORT CONNECT
- LISTCAT LEVEL, and other catalog commands that list the catalog in a generic manner.

**STGADMIN.IGG.DLVVRNVR.NOCAT**
protects the ability to delete a VVR or NVR without an associated catalog. Users having RACF READ authority to the facility class will need no other RACF authority to the master catalog to perform the DELETE NVR or DELETE VVR functions.

**STGADMIN.IGG.LIBRARY**
protects the ability to DEFINE, DELETE or ALTER tape library and tape volume entries.

Access to this facility class should be restricted to users who understand the risk involved in deleting a VVR or NVR entry from a VVDS.

# RACF-Controlled ERASE Options

DELETE processing removes catalog and VTOC information; it also makes the associated DASD space available for a new allocation. By default, this process does not erase the data from the disk. Data is erased by overwriting it with binary zeros. Sensitive data should be erased before its space is made available.

You can use RACF commands to specify an ERASE or NOERASE attribute in generic or discrete profiles. When so specified, these attributes become default attributes for:

- DELETE processing of an alternate index or cluster cataloged in an integrated catalog facility catalog.
- Scratch and partial release processing of non-VSAM data sets (see *z/OS DFSMSdfp Advanced Services* for more information).

See "Deleting Sensitive Data" on page 71 for more information on deleting sensitive data.

## Access Method Services Password Protection

Passwords are no longer honored for protecting an ICF catalog, or data sets contained in an ICF catalog. Previously, passwords were ignored only for SMS-managed data sets. You should use RACF to protect your data.

**Note:** If an ICF catalog is shared with a downlevel system, data sets will remain password protected in the downlevel system, but not in the DFSMS/MVS™ Version 1 Release 5 system.

If you are transferring data from a system with RACF to a system which does not have RACF, data sets in an ICF catalog will not be protected.

# Chapter 6. Backing Up and Recovering Catalogs

Because catalogs are essential system data sets, it is important that you maintain backup copies. The more recent and accurate a backup copy, the less impact a catalog outage will have on your installation.

Besides the backup and recovery of BCSs, you should also develop a strategy for backing up VVDSs, VTOCs, and VTOC indexes.

This chapter considers the backup and recovery of BCSs and VVDSs, and indirectly, the VTOC. For more information about data set back up and recovery, see the following:

- *z/OS DFSMSdfp Storage Administration Reference*
- *z/OS DFSMS: Using Data Sets*
- *z/OS DFSMSdss Storage Administration Guide*
- *z/OS DFSMShsm Data Recovery Scenarios*

## Developing a Backup and Recovery Strategy

A primary consideration for backing up catalogs is backup frequency. In general, a catalog recovery takes less time the more recently the backup copy was taken. However, continuously creating backup copies can be a drain on your system. Daily catalog backup should be sufficient for most catalogs.

Backup procedures can be simplified if you are using the Storage Management Subsystem. You can define a management class for your catalogs and other system data sets, defining an appropriate backup frequency. The system then creates backups of the catalog according to your management policy.

Your backup strategy should include running IDCAMS DIAGNOSE and EXAMINE functions against each catalog before you perform a back up. This ensures that you back up only valid catalogs.

The BCS can be backed up as a data set using the EXPORT command, DFSMSdss, or DFSMShsm. The aliases defined for the catalog are saved with the backup copy when EXPORT or DFSMShsm is used, or when logical dump is used with DFSMSdss. When you recover the catalog by importing it, you can have the saved aliases redefined and merged with existing aliases. DFSMSdss and DFSMShsm redefine the aliases automatically. However, when you use the IMPORT command, you must specify ALIAS to have aliases redefined or retained.

Before recovering a BCS, you should lock the catalog to prevent access during the recovery. Before you can lock the catalog, no job can have the catalog allocated with a disposition of OLD. This disposition can be defined on a JOBCAT, STEPCAT, or other DD statement. It can also result if the catalog is used in the OUTDATASET parameter of access method services commands. The RACF FACILITY class profile IGG.CATLOCK must be defined to allow the use of the LOCK parameter with ALTER, DEFINE USERCATALOG, and IMPORT to those users having READ authority to this class. All other access is restricted from LOCK use.

The VVDS and VTOC should not be backed up as data sets, but are backed up as part of a full volume dump using DFSMSdss or DFSMShsm. The entries in the

VVDS and the VTOC are backed up with the data sets they describe when the data sets are backed up with the IDCAMS EXPORT command, DFSMShsm, or DFSMSdss logical dump.

There are two ways that a VVDS or VTOC can be recovered:
1. Restore the volume containing the VVDS or VTOC, or
2. Rebuild the VVDS and VTOC by recovering the data sets on the volume.

Restoring the volume is the easiest way to recover a VVDS or VTOC. However, this is seldom practical because the data sets restored will not be current. To rebuild the VVDS, you must delete it and then recover all VSAM and SMS-managed data sets which were on the volume.

The VTOC can be backed up with volume dumping, and can be restored by restoring the volume. To rebuild a VTOC, you have to use ICKDSF to initialize the volume. Then, all data (not only VSAM data sets), must be recovered to the volume. Do not try to repair a VTOC by manually rebuilding damaged records.

BCS backups and volume dumps can be stored on DASD or tape. More than one backup should be kept, since the most recent backup might also be damaged.

See *MVS/ESA SML: Managing Data* and *MVS/ESA SML: Managing Storage Groups* for a more extensive discussion of backup and recovery considerations.

# Backing Up a Catalog

The two parts of a catalog, the BCS and the VVDS, require different backup techniques: the BCS can be backed up like other data sets, whereas the VVDS can only be backed up as part of a volume dump. The entries in the VVDS and VTOC are backed up when the data sets they describe are exported with access method services, logically dumped with DFSMSdss, or backed up with DFSMShsm.

# Backing Up a BCS

You can use the access method services EXPORT command, the DFSMSdss logical DUMP command, or the DFSMShsm BACKDS command to back up a BCS. You can later recover the backup copies using the same utility used to create the backup: the access method services IMPORT command for exported copies; the DFSMSdss RESTORE command for logical dump copies; and the DFSMShsm RECOVER command for DFSMShsm backups.

The copy created by these utilities is a "portable" sequential data set that can be stored on a tape or direct access device, which can be of a different device type than the one containing the source catalog.

When these commands are used to back up a BCS, the aliases of the catalog are saved in the backup copy. The source catalog is not deleted, and remains as a fully functional catalog. The relationships between the BCS and VVDSs are unchanged.

You cannot permanently export a catalog by using the PERMANENT parameter of EXPORT. The TEMPORARY option is used even if you specify PERMANENT or allow it to default.

To ensure the integrity of the copy, access to the BCS is serialized by these commands. This serialization prevents update access but allows read access from the system performing the command. If you are using multi-system global resource serialization (GRS) or equivalent product to convert the catalog RESERVE to a

GLOBAL ENQUEUE, other sharing systems can also have read access to the catalog. Otherwise, the RESERVE (with SYSIGGV2) makes the catalog inaccessible from other systems for the duration of the command.

These commands do not back up corresponding entries in any related VVDS except those describing the BCS itself.

For more information on using DFSMSdss, see *z/OS DFSMSdss Storage Administration Reference*. For more information on using DFSMShsm, see *z/OS DFSMShsm Storage Administration Guide*.

**Restriction:** You cannot use IDCAMS REPRO or other copying commands to create and recover BCS backups.

## Backing Up a Master Catalog

A master catalog can be backed up as any other BCS. You should use EXPORT, DFSMSdss, or DFSMShsm for the backup. Another way to provide a backup for the master catalog is to create an alternate master catalog. For information on defining and using an alternate master catalog, see "Creating and Using an Alternate Master Catalog" on page 24.

You should also make periodic volume dumps of the master catalog's volume. This dump can later be used by the stand-alone version of DFSMSdss to restore the master catalog, if you cannot access the volume from another system.

## Backing up a VVDS

The VVDS should not be backed up as a data set to provide for recovery. To back up the VVDS, back up the volume containing the VVDS, or back up all data sets described in the VVDS (all VSAM and SMS-managed data sets). If the VVDS ever needs to be recovered, recover the entire volume, or all the data sets described in the VVDS.

You can use either DFSMSdss or DFSMShsm to back up and recover a volume or individual data sets on the volume. For further information, see *z/OS DFSMSdss Storage Administration Reference* and *z/OS DFSMShsm Storage Administration Guide*.

## Recovering a Catalog

Normally, a BCS is recovered separately from a VVDS. A VVDS usually does not need to be recovered, even if an associated BCS is recovered. However, if you need to recover a VVDS, and a BCS resides on the VVDS's volume, you must recover the BCS as well. If possible, you should export the BCS before recovering the volume, and then recover the BCS from the exported copy. This ensures a current BCS.

Before recovering a BCS or VVDS, try to recover damaged records. If damaged records can be rebuilt, you can avoid a full recovery. BCS records can be recovered using access method services DELETE and DEFINE commands with appropriate parameters. VVDS and VTOC records can be recovered using the DELETE command and by recovering the data sets on the volume. See "Updating the Catalog After Recovery" on page 90 for more information on recovering individual entries.

# Locking a Catalog

You should restrict access to a catalog when you are recovering it or when you are performing other maintenance procedures which involve redefining the catalog. If you do not restrict access to the catalog (by locking it, by terminating user sessions, or by another method), users might be able to update the catalog during recovery or maintenance and create a data integrity exposure. Locking the catalog eliminates the need to terminate user sessions during catalog recovery or maintenance.

You can only lock integrated user catalogs. You cannot lock a master catalog. While the catalog is locked, unauthorized requests to access the catalog fail with a return code indicating that the catalog is temporarily unavailable. Jobs entered with JCL fail with a JCL error. You cannot make JCL jobs wait until the catalog is unlocked. The catalog is also unavailable to any system that shares the catalog.

After you have completed the catalog recovery or maintenance, unlock the catalog so that normal operations can resume.

To lock or unlock a catalog, you use the LOCK and UNLOCK parameters of the access method services ALTER, DEFINE, or IMPORT commands. You use the ALTER command to lock or unlock an existing catalog; the DEFINE command to lock a newly defined catalog (the default is to define an unlocked catalog); and the IMPORT command to lock or unlock a catalog that you are importing. In order to lock or unlock a catalog, you must have READ access to the IGG.CATLOCK profile in RACF, and ALTER authority to the catalog. If the catalog is shared between systems, you may want to ensure you provide access to those users who may need to access the catalog while it is locked (such as VTAM™, described below).

**Exception:** Catalogs are not unlocked during a system IPL. If you lock a catalog, and there is a system failure, the catalog is still locked after you IPL the system. This can cause problems if a locked catalog contains entries for data sets needed during IPL.

For example, if the catalog containing entries needed for VTAM is locked, VTAM cannot be started. Because VTAM is needed to start TSO, and TSO must be active to issue the ALTER command or submit a batch IDCAMS job, you cannot use ALTER UNLOCK to unlock the catalog.

As long as TSO is available, you can simply use ALTER UNLOCK to unlock the catalog and allow the IPL to complete. However, you can also authorize VTAM to the IGG.CATLOCK profile. This allows VTAM to access a locked catalog. If you authorize VTAM to IGG.CATLOCK, you should also authorize any other components which are needed to start VTAM.

Jobs such as VTAM and any other critical system resource should be given CATLOCK authority prior to locking any user catalog. If TSO is not available, and VTAM cannot be started because it does not have access to the IGG.CATLOCK profile, you must use a card reader to enter an IDCAMS ALTER UNLOCK job into the system to unlock the catalog.

If the catalog is shared between systems, it may be unlocked from any of the shared systems. Thus the catalog does not need to be unlocked from the system that locked it. This provides an alternative way to recover when the system that locked the catalog cannot be used to unlock it.

# Recovering a BCS

You can recover a BCS that was backed up with the access method services EXPORT command, the DFSMSdss logical DUMP command, or the DFSMShsm BACKDS command or automatic backup. To recover the BCS, use the IDCAMS IMPORT command, the DFSMSdss RESTORE command, or the DFSMShsm RECOVER command.

When you recover a BCS using these commands, you do not need to delete and redefine the target catalog unless you want to change the catalog's size or other characteristics, or unless the BCS is damaged in such a way as to prevent the usual recovery. The recovered catalog is reorganized when you use IMPORT or RECOVER, but not when you use RESTORE.

Aliases to the catalog can be defined if you use DFSMSdss, DFSMShsm, or if you specify ALIAS on the IMPORT command. If you have not deleted and redefined the catalog, all existing aliases are maintained, and any aliases defined in the backup copy are redefined if they are not already defined.

If you do not first delete the catalog you are recovering (specifying RECOVERY), the catalog is deleted and redefined according to the attributes of the backup copy. If you delete and redefine the catalog before you recover it (and the newly defined catalog is empty), the backup copy is copied into the new catalog. Specify INTOEMPTY on the IMPORT command if you define the catalog before importing it.

When using IMPORT with the OUTFILE parameter, the DD statement defining the catalog must have a disposition of OLD.

Before you recover a BCS, lock the BCS. After you recover the catalog, update the BCS with any changes which have occurred since the last backup. You can use the access method services DIAGNOSE command to identify certain unsynchronized entries.

To simplify catalog recovery and improve data availability for your installation, use the Integrated Catalog Forward Recovery Utility. (Program number 5798-DXQ) to update the BCS to a current status. For more information, see *ICFRU Program Description/Operations.*

The following are the steps needed to recover a BCS using the IMPORT command. For further discussion of using DFSMSdss and DFSMShsm in BCS recovery, see *z/OS DFSMSdss Storage Administration Guide* and *z/OS DFSMShsm Data Recovery Scenarios*. The catalog being recovered is SYS1.ICFCAT.PROJECT1.

1. If the catalog is used by the job scheduler for any batch jobs, hold the job queue for all job classes except the one you use for the recovery. Interactive users are returned appropriate messages if a catalog is locked and they try to use it. However, batch jobs fail with a JCL error if they try to use a locked catalog.

2. Lock the catalog so that access is restricted. Note that in order to lock or unlock a catalog, you must have READ access to the IGG.CATLOCK profile in RACF, and ALTER authority to the catalog. See "Locking a Catalog" on page 84 for more information.

```
//LOCKCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
     ALTER SYS1.ICFCAT.PROJECT1 LOCK
/*
```

3. Use the Integrated Catalog Forward Recovery Utility to create an updated EXPORT backup copy of the BCS, using the last backup copy and the appropriate SMF records.

4. Import the most current backup copy of the BCS (which contains the BCS's aliases as they existed when the backup was made) using the access method services command IMPORT ALIAS LOCK. The most current backup is the one created in step 3. If the aliases are not needed, do not specify the ALIAS parameter. If you do not specify LOCK, the catalog is unlocked.

```
//RECOVER  EXEC PGM=IDCAMS
//BACKCOPY DD DSN=BACKUP.SYS1.ICFCAT.PROJECT1,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
    IMPORT INFILE(BACKCOPY) -
         OUTDATASET(SYS1.ICFCAT.PROJECT1) -
         ALIAS -
         LOCK

 /*
```

5. If you did not use the Integrated Catalog Forward Recovery Utility to create an updated backup copy of the BCS, you need to manually check for recent changes to the BCS which are not reflected in the recovered copy.

   Use the SMF records which record changes to the catalog, and any tape management records you keep. An DFSMShsm audit might also be helpful.

   The backup or portable copy of a catalog reflects the contents of the catalog at the time it was backed up with EXPORT, DFSMSdss, or DFSMShsm. Any subsequent ALTER, DEFINE or DELETE operations are not reflected in the catalog when it is imported.

   The access method services command DIAGNOSE can be used after importing the catalog, to help assess activity not reflected in the imported catalog. However, DIAGNOSE is only useful for VSAM or SMS-managed data sets on DASD. Tape data sets cannot be processed by DIAGNOSE. List the VVDSs connected to the catalog to determine which VVDSs to compare with the BCS.

   If you were able to list the non-VSAM data sets before recovering the catalog, compare the list taken before the recovery with one taken after the recover. Make any needed changes by following the procedures in "Updating the Catalog After Recovery" on page 90.

   For complete information on using DIAGNOSE, see "Analyzing a Catalog for Synchronization Errors" on page 95.

6. Use the EXAMINE and DIAGNOSE commands to check the integrity of the recovered catalog's structure and content. If EXAMINE indicates structural errors, you need to recover the next most recent backup copy, repeating these steps. If DIAGNOSE indicates errors, address the errors as appropriate.

```
//EXAMINE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
   EXAMINE NAME(SYS1.ICFCAT.PROJECT1) -
         INDEXTEST -
         DATATEST
 /*
```

7. If you recovered the catalog onto a volume with a different volume serial number or device type (for example, if the catalog was damaged due to volume damage), and the catalog is shared with other systems, use IMPORT CONNECT ALIAS to update the catalog connector records in the master catalogs of the sharing systems. See "Recovering Shared Catalogs" on page 87 for more information on recovering shared catalogs.

8. Unlock the catalog with the command ALTER UNLOCK to allow general access.

```
//UNLCKCAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
     ALTER SYS1.ICFCAT.PROJECT1 UNLOCK
/*
```

9. Free the job queue if you put it on hold.

# Recovering Shared Catalogs

When two or more systems share access to a user catalog and the user catalog is recovered on one of those systems to a different volume or device type, you will need to update or replace the catalog connector record in the master catalogs on the other systems. Otherwise, the recovery considerations for shared catalogs are the same as those described in "Recovering a BCS" on page 85.

The best way to update the catalog connector record on the sharing systems is to use the access method services IMPORT CONNECT ALIAS command. This command maintains any aliases already defined for the catalog. You can use this command even if the catalog has no aliases defined for it.

If you want to delete the aliases for a catalog, you can use EXPORT DISCONNECT followed by IMPORT CONNECT.

For example, the user catalog SYS1.ICFCAT.SHARED, which has many aliases, resides on volume 339001 (a 3390), and is shared by SYSTEMA and SYSTEMB. If SYS1.ICFCAT.SHARED is successfully recovered by SYSTEMA to volume 339002 (another 3390), SYS1.ICFCAT.SHARED is inaccessible to SYSTEMB because its connector record in SYSTEMB's master catalog has an incorrect volume serial number. Executing the following step on SYSTEMB updates the volume serial number and preserves the aliases already defined for the catalog:

```
//CONNECT  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
    IMPORT CONNECT ALIAS -
        OBJECTS((SYS1.ICFCAT.SHARED  -
                  DEVICETYPE(3390)   -
                   VOLUMES(339002)))
/*
```

If you use EXPORT DISCONNECT, and IMPORT CONNECT, all aliases for the user catalog are deleted from SYSTEMB and must be redefined if needed.

## RLS Considerations When Recovering Shared Catalogs

If you use VSAM record-level sharing (RLS), the AMS SHCDS CFREPAIR command should be used. When a catalog is recovered, but before making the catalog available, use the AMS SHCDS CFREPAIR command to reconstruct critical RLS information in the catalog. The catalog must be import connected on all systems to the master catalog before the SHCDS CFREPAIR command can be used (see "Recovering Shared Catalogs"). See *z/OS DFSMS Access Method Services* for more information about the SHCDS command.

# Recovering a Master Catalog

A master catalog cannot be recovered while it is being used as a master catalog. To recover the catalog, you must first make it a user catalog.

This can be accomplished in either of two ways:

1. IPL the system using an alternate master catalog. The old master catalog should be identified as a user catalog in the alternate master catalog. After you recover the old master, you can IPL the system using the recovered catalog.
2. Recover the catalog from a shared system. The catalog is defined as a user catalog in the other system's master catalog, and could be addressed on the sharing system as a user catalog.

You can also recover the master catalog using DFSMSdss if you have a physical dump of the volume containing the master catalog.

## Recovering an Unavailable Catalog

If a catalog becomes unavailable, that is, it cannot be opened and accessed correctly, you should determine the cause of the problem before attempting a recovery.

The following sequence of system messages indicates that the system was unable to open the catalog:

```
IEC331I   return code 004    reason code 40
IEC161I   return code 004    reason code 80
IEC331I   return code 004    reason code 86
```

These messages are usually accompanied by an IDC3009I message.

Before recovering the catalog, determine if the problem is caused by a broken connection between the catalog and the master catalog. Use LISTCAT ALL to look at the entry for the catalog, comparing the volume serial number and device type listed to the actual ones. If the catalog connector record is inaccurate or missing, use IMPORT CONNECT ALIAS to connect the BCS to the appropriate master catalog.

These messages might also indicate that the VVDS is unavailable. If so, recover the VVDS.

If the catalog is properly connected to the master catalog and the VVDS is available, then recover the catalog. You might need to delete the catalog's record in the VVDS, and its DSCB in the VTOC, before importing a backup copy. The access method services DELETE USERCATALOG RECOVERY command removes the VVDS and VTOC entries for the catalog.

## Recovering a VVDS

Before recovering a VVDS, decide if the VVDS is systematically damaged, or if only certain entries in the VVDS are damaged. If you cannot open the VVDS, for example, when you try to print it or access data sets which have entries in it, then the VVDS is probably systematically damaged, and should be recovered in its entirety.

If you can open the VVDS, run DIAGNOSE to determine which entries are damaged. You can then use the access method services DELETE command followed by data set recovery to recreate the VVDS entries for the affected data sets, and avoid a total VVDS recovery.

If you decide you must recover the VVDS in its entirety, then all data sets represented in the VVDS must be recovered, either individually or by a full-volume restore. Use DFSMSdss or DFSMShsm for volume recovery.

If you are not using SMS, then only VSAM data sets are affected. Otherwise, all data sets on the volume are affected, and must be recovered.

Before recovering a volume, it is necessary to get the volume offline, so that users cannot allocate resources on the volume as you try to restore it. Use the following procedure to get the volume offline if it is not managed by the Storage Management Subsystem:

1. Use the VARY command to get the volume offline.
2. Use the DISPLAY command to determine if the volume has been successfully varied offline, or if resources are still allocated on the volume.
3. Use MODIFY CATALOG to unallocate the VVDS or any catalogs on the volume which are allocated. Use the VUNALLOCATE parameter to unallocate the VVDS. Use the UNALLOCATE command to unallocate the catalog.

If the volume is SMS-managed, set the SMS VOLUME STATUS to DISALL before using the VARY command. Then, check for allocations with the DISPLAY command, and use MODIFY CATALOG if necessary.

After you have gotten control of the volume, use DFSMShsm or DFSMSdss to recover it. For information on using the DFSMShsm RECOVER command to recover a volume with incremental backup copies, see *z/OS DFSMShsm Data Recovery Scenarios*. For information on volume recovery with DFSMSdss, see *z/OS DFSMSdss Storage Administration Reference*.

# Recovering Tape Volume or Tape Library Entries

Access method services cannot change the library manager inventory in an automated tape library. ISMF should be used for normal tape library create functions. The access method services CREATE LIBRARYENTRY or CREATE VOLUMEENTRY commands should be used only to recover from tape volume catalog errors.

### Creating a Tape Library Entry:
The CREATE LIBRARYENTRY command creates a tape library entry.

This example creates an entry for a tape library named ATLLIB1.

```
//CREATLIB JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
    CREATE LIBRARYENTRY -
          (NAME(ATLLIB1) -
          LIBRARYID(12345) -
          LIBDEVTYPE(3495) -
          LOGICALTYPE(AUTOMATED) -
          NUMBERSLOTS(15000) -
          NUMBEREMPTYSLOTS(1000) -
          NUMBERSCRATCHVOLUMES(MEDIA1(500) MEDIA2(400)) -
          SCRATCHTHRESHOLD(MEDIA1(200) MEDIA2(100)) -
          DESCRIPTION('TEST LIBRARY ATLLIB1') -
          CONSOLENAME(TESTCON)
```

### Creating a Tape Volume Entry:

The CREATE VOLUMEENTRY command creates tape volume entries.

This example creates a tape library entry for a volume with volser AL0001.

```
//CREATVOL JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
     CREATE VOLUMEENTRY -
          (NAME(VAL0001) -
          LIBRARYNAME(ATLLIB1) -
          STORAGEGROUP(*SCRTCH*) -
          USEATTRIBUTE(SCRATCH) -
          NOWRITEPROTECT -
          LOCATION(LIBRARY) -
          SHELFLOCATION(10098SHELF) -
          OWNERINFORMATION('JOHN SMITH,RMKD222') -
          ENTEREJECTDATE(1990-03-18) -
          EXPIRATIONDATE(2000-12-31) -
          WRITEDATE(1991-01-02) -
          MOUNTDATE(1991-01-02))
/*
```

# Updating the Catalog After Recovery

After you recover a catalog, use the access method services DIAGNOSE command to help determine how the recovered catalog differs from the current status of data sets, VVDSs, and BCSs on your system. Some activities that alter data sets will change the information in the BCS or VVDS records, and you will need to update your catalog to reflect any changes which have occurred since the catalog was last backed up.

The DIAGNOSE command identifies differences between BCS and VVDS records for VSAM and SMS-managed non-VSAM data sets. To identify inaccurate BCS records for other non-VSAM data sets, use LISTCAT NONVSAM, if possible, before and after the recovery.

The access method services DEFINE and DELETE commands can be used to update the catalog. With these commands, you can: recatalog a data set, thus updating the information in the catalog records; delete a data set; delete a data set's record in the BCS or VVDS; and delete a data set's DSCB in the VTOC.

Table 10 shows the activities that might have occurred since the last catalog backup, and the tasks required to update the catalog.

*Table 10. Activities That Downgrade a Basic Catalog Structure (BCS)*

| Activity Causing Downgrading | Data Set Type | Location of Information | Action Needed To Update the Catalog |
|---|---|---|---|
| **Add** | | | |
| Records | N | VTOC/tape labels | None |
| | V | VVDS | |
| Extents | N | VTOC/tape labels | None |
| | V | VTOC/VVDS | |
| Volumes | N | BCS/VTOC/tape labels [1] | Recatalog the data set [2] |
| | V | BCS/VTOC/VVDS | |

| Activity Causing Downgrading | Data Set Type | Location of Information | Action Needed To Update the Catalog |
|---|---|---|---|
| Candidate Volumes | V | BCS | Add the volume with ALTER ADDVOLUMES |
| Data Sets | N<br>V | BCS/VTOC/tape labels [1]<br>BCS/VTOC/VVDS | Recatalog the data set [2] |
| **Remove** | | | |
| Records | N<br>V | VTOC<br>VVDS | None |
| Extents | N<br>V | VTOC<br>VTOC/VVDS | None<br>Not applicable to VSAM |
| Volumes | N<br>V | BCS/VTOC [1]<br>BCS/VTOC/VVDS | Recatalog the data set [2]<br>Not applicable to VSAM |
| Candidate Volumes | V | BCS | Remove volume with ALTER REMOVEVOLUMES |
| Data Sets | N<br>V | BCS/VTOC/tape labels [1]<br>BCS/VTOC/VVDS | DELETE NOSCRATCH to remove BCS record |
| **Modify** | | | |
| Records | N<br>V | (data set only)<br>VVDS | None |
| Extents | N<br>V | VTOC<br>VTOC/VVDS | None<br>Not applicable to VSAM |
| Volumes | N<br>V | BCS/VTOC/tape labels [1]<br>BCS/VTOC/VVDS | Recatalog the data set [2]<br>Not applicable to VSAM |
| Data Sets | N<br>V | BCS/VTOC [1]<br>BCS/VTOC/VVDS | Recatalog the data set [2] |

**Notes:**

**N**     Non-VSAM data set
**V**     VSAM data set

1. SMS-managed non-VSAM data sets also have VVDS entries.

2. To recatalog a data set, first use DELETE NOSCRATCH to remove BCS record for the data set. Then, use DEFINE to recreate the entry. If the data set is VSAM or SMS-managed, specify RECATALOG on the DEFINE command, so that the BCS record is rebuilt using information from the VVDS.

# Recataloging Data Sets and VSAM Objects

After a BCS recovery, some entries in the recovered BCS might not accurately reflect the current characteristics of a data set or VSAM object. The VVDS, VTOC, and tape labels should contain the accurate information for existing data sets. Data sets can only be recataloged into the catalog specified in the VVR/NVR unless they are pagespace, swapspace or SYS1 data sets.

For SMS-managed data sets and all VSAM data sets and associations, BCS entries can be recataloged using the access method services DEFINE command with the RECATALOG option. If there is a BCS entry for the data set, first remove it using DELETE NOSCRATCH.

For non-VSAM, non-SMS-managed data sets, you can delete the BCS entry using the DELETE NOSCRATCH command. Then, you can catalog the data set with DEFINE NONVSAM. If the BCS entry for the data set is missing, you can use DEFINE NONVSAM without first using DELETE. For information on DEFINE NONVSAM, see *z/OS DFSMS Access Method Services*.

To recatalog a data set or VSAM object using the DEFINE RECATALOG command, the corresponding entries in the VVDS and VTOC must exist. This information is used to rebuild the BCS entries, along with information supplied in the command. Data sets can only be recataloged into the catalog specified in the VVR/NVR unless they are pagespace, swapspace or SYS1 data sets.

You might also need to recatalog a data set or VSAM object if you used DELETE with the NOSCRATCH option to delete a BCS entry, or if you restored a volume.

To successfully recatalog a data set or VSAM object, you generally must supply the entry's name, volume, and any ownership, protection, or expiration attributes defined for the entry. Occasionally, you might need to specify the space attribute, and the data set organization. When recataloging alternate indexes, you must also specify the object to which the index is related.

When recataloging VSAM data sets, do not specify component names. They are obtained from the VVDS.

If the data set or VSAM object is protected by a discrete RACF profile, it must be recataloged with that profile. This information is kept in the VTOC or VVDS.

## Recataloging a VVDS

After recovery, a BCS might not contain entries for all the VVDSs on volumes where the BCS has data sets. In this case, you might want to recatalog the VVDS so that the BCS contains entries for all connected VVDSs.

If you want to recreate the BCS entry for a VVDS, use the access method services DEFINE CLUSTER command with the RECATALOG option. Specify the name, volume of the VVDS, and NONINDEXED. The BCS entry is rebuilt using information in the VVDS and the command. A VTOC entry for the VVDS must also exist.

## Deleting BCS Records

When you delete a data set, the BCS, VVDS, and VTOC entries for the data set are removed. If you later recover a BCS, there might be BCS entries for data sets which have been deleted. In this case, the data sets do not exist, and there are no entries for them in the VVDS or VTOC. To clean up the BCS, delete the BCS entries.

For non-VSAM data sets and VSAM clusters, alternate indexes, and page spaces, the DELETE command with the NOSCRATCH option removes the BCS entries. If you define the appropriate RACF FACILITY class, only authorized users are allowed to use NOSCRATCH on SMS-managed data sets.

When you use the NOSCRATCH option, the VVDS and VTOC are not changed. Using this option is the same as uncataloging a data set. After deleting the BCS entry, you can recatalog the data set.

The BCS record for a VVDS might only be deleted if there are no records in the VVDS for data sets cataloged in the BCS. When the BCS record for a VVDS is deleted, the back-pointer to the BCS in the VVDS is also deleted.

You might also need to delete VSAM truename records. VSAM data sets and objects which have more than one component have more than one BCS entry. For example, a key-sequenced data set has entries for the data component and the index component ("truename" records), as well as an entry for the data set itself ("sphere" record).

In order to use a VSAM data set or object, there must be a sphere record for it. If that record is missing, delete the truename records for the data set's components using DELETE TRUENAME. Then recatalog the data set with the DEFINE command. If the sphere record is accessible, then the sphere record and associated truename records can be deleted using the NOSCRATCH option of DELETE.

You can only delete a truename record if the associated sphere record is missing or inaccessible. This problem can be identified using the DIAGNOSE command. You cannot define a data set with the same name as a data set with a missing sphere record: the data set's name cannot be reused until the truename records are deleted.

## Deleting VVDS Records and VTOC DSCBs

Occasionally, a VVDS might have a record for a data set which no longer exists, or for which there is no corresponding BCS entry. The VVDS contains VVRs for all VSAM data sets and objects, and on SMS-managed volumes, NVRs for non-VSAM data sets.

You can use the access method services DELETE command to clean up a VVDS and to remove records for data sets which no longer exist or which have become uncataloged. Specify either VVR or NVR to delete the appropriate type of record.

The DELETE command only deletes the VVR or NVR if there is no corresponding entry in a BCS for the data set. When the VVDS record is deleted, the corresponding DSCB in the VTOC is also scratched. If there is no VVDS record for the data set, the DSCB is still scratched. In order to delete a VVR, specify the component name (the "truename") to be deleted, rather than the cluster name.

STGADMIN.IGG.DLVVRNVR.NOCAT protects the ability to delete a VVR or NVR without an associated catalog. See topic 78 for more information.

## Recovering Data Sets

After recovering a volume or data set, you might have to recover some individual data sets. The sequence of commands you use should depend on the existence of entries for the data set in the BCS, VVDS, and VTOC. Normally, you only have to restore the latest backup copy of the data set. In some circumstances, you should first delete the data set's catalog or VTOC entries before recovery.

The following table shows the possible relationships between entries in the BCS, VVDS, and VTOC for a single data set. The recovery procedures noted are further explained after the table.

| BCS | VVDS | VTOC | Recovery Action |
|---|---|---|---|
| entry present | entry present | entry present | normal |
| entry present | entry present | entry missing | normal |
| entry present | entry missing | entry present | normal |
| entry present | entry missing | entry missing | normal |
| entry missing | entry present | entry present | clean up, then recover |
| entry missing | entry present | entry missing | clean up, then recover |
| entry missing | entry missing | entry present | clean up, then recover |
| entry missing | entry missing | entry missing | normal |

Following is an explanation of the appropriate recovery procedures:

**normal**

In these cases, you do not need to perform special actions before recovering a data set. For VSAM data sets and objects, import an exported copy. For non-VSAM data sets, replace the data set with a backup copy.

**clean up, then recover**

In these cases, where there are VVDS or VTOC entries for a data set, but no BCS entry, first delete the VVDS or VTOC entries. Then recover the data set using the normal procedure.

If you do not first clean up the VVDS, importing a VSAM data set or restoring an SMS-managed data set may create duplicate VVDS records. This duplication may cause problems when you try to use the data set.

If there is only a DSCB for a VSAM data set, you cannot import the data set unless you scratch the DSCB. For an explanation of the procedure for deleting VVDS records and VTOC DSCBs, see "Deleting VVDS Records and VTOC DSCBs" on page 93.

# Chapter 7. Analyzing Catalogs for Errors and Synchronization

This chapter describes how you can analyze your catalogs of DASD data sets. Proper catalog functioning requires that the structure of the BCS remain sound. It also requires that data set information contained in the BCS, VVDS, and VTOC be synchronized. That is, the characteristics of a data set (extent information, SMS class names, etc.) must be the same in the data set's entries in the BCS, VVDS, and VTOC.

Structurally damaged or unsynchronized catalogs can cause jobs which use the catalog to fail or produce incorrect results if the wrong data set is processed. If the number of corrupt entries becomes large, you might have to recover the catalog.

It is strongly recommended that you perform the DIAGNOSE and EXAMINE functions regularly for all catalogs in your installation. They can be combined with regular backup functions, for example. Performing these functions regularly ensures that backups taken of catalogs are structurally valid and also helps identify, as early as possible, when a catalog has possibly been damaged.

## Analyzing a BCS for Structural Errors

If a catalog is causing jobs to fail and you cannot trace the failure to unsynchronized entries in the catalog, the catalog might be structurally unsound. Structural problems affect BCS characteristics as a VSAM key-sequenced data set, not as a catalog.

You must have READ authority for the RACF STGADMIN.IDC.*xxx* profile, if it is defined. For more information about STGADMIN RACF profiles, see "Storage Administration (STGADMIN) Profiles in the FACILITY Class" on page 76.

To test the structure of a BCS, perform the following steps:
1. Execute the access method services VERIFY command to verify that the VSAM information for the catalog is current. If the catalog is currently open in the catalog address space and the EXAMINE command is executed, the data that is contained in the catalog may not be current.
2. Execute the access method services ALTER command to lock the catalog and prevent updates to the catalog while you are inspecting its structure.
3. Execute the access method services EXAMINE command. With this command, you can test both the index and data components of a BCS.
4. If the BCS has structural errors, recover the catalog with the most recent structurally sound backup copy.

For more information on using EXAMINE, see *z/OS DFSMS Access Method Services* and *z/OS DFSMS: Using Data Sets*.

## Analyzing a Catalog for Synchronization Errors

Catalog entries might become unsynchronized, so that information about the attributes and characteristics of a data set are different in the BCS, VVDS, and VTOC. These differences may make a data set inaccessible or otherwise unusable.

To analyze a catalog for synchronization errors, you can use the access method services DIAGNOSE command. With this command, you can analyze the content of catalog records in the BCS and VVDS, and compare VVDS information with DSCB information in the VTOC.

Besides checking for synchronization errors, DIAGNOSE also checks for invalid data, or invalid relationships between entries.

To use DIAGNOSE ICFCATALOG, you must have READ authority to the RACF STGADMIN.IDC.DIAGNOSE.CATALOG profile if it is defined. DIAGNOSE VVDS requires READ authority to STGADMIN.IDC.DIAGNOSE.VVDS. See "Storage Administration (STGADMIN) Profiles in the FACILITY Class" on page 76 for more information on STGADMIN profiles.

If you are using DFSMShsm, catalog synchronization is checked during the daily data management cycle, and synchronization errors are reported. This eliminates the need to use DIAGNOSE separately, unless you are recovering a catalog. Whenever you recover a catalog, use DIAGNOSE to help identify catalog records which must be updated or which are otherwise in need of individual recovery. Updating catalog records after recovery is discussed in "Updating the Catalog After Recovery" on page 90.

# Using the DIAGNOSE Command

You can use the DIAGNOSE command to accomplish two main tasks:
1. Check the dependent content of catalog records
2. Check the validity of the content of catalog records.

There are no special parameters needed for simply checking the content of a record. As with all DIAGNOSE jobs, specify the ICFCATALOG parameter when checking a BCS, and the VVDS parameter when checking a VVDS.

DIAGNOSE recognizes tape library and tape volume record types. DIAGNOSE checks the cell structure of the volume catalog.

To compare catalog information with VTOC information, you must use DIAGNOSE VVDS, to analyze a VVDS. When you use DIAGNOSE ICFCATALOG, only the BCS and related VVDSs are analyzed.

All records of the input BCS or VVDS are processed unless they are explicitly included or excluded.

## Checking the Dependent Content of a Record

To check the dependent content of catalog records, use the COMPAREDD or COMPAREDS parameters. If you are analyzing a BCS, specify related VVDSs. If you are analyzing a VVDS, specify related BCSs. You can limit dependency checking to selected BCSs or VVDSs, rather than specifying all related BCSs or VVDSs.

For example, the VVDS is checked for an entry that is consistent with the BCS entry. This is not a complete check of the external data set or its entry. It is only a consistency check between the two.

If you are comparing:
• A BCS, the VVDS record is checked for dependency.
• A VVDS, the BCS record, and the VTOC DSCB are checked for dependency.

The input BCS or VVDS determines which entries in the BCSs or VVDSs are used for comparison. Only entries referenced in the input BCS or VVDS are checked.

If dependency checking is not specified, the external pointers to the BCS are not used or checked. The DIAGNOSE command does not identify VVDSs which should be specified for dependency checking.

To determine which VVDSs are connected to a BCS, use LISTCAT LEVEL(SYS1.VVDS), and specify the BCS in the CATALOG parameter. To determine which BCSs are connected to a VVDS, use PRINT COUNT(1) to print the first record of the VVDS, which contains the names of the related BCSs.

## Limiting the Scope of DIAGNOSE

You can specify which records are to be analyzed, or you can exclude certain records from analysis. In this way, you can limit the analysis to records you suspect are in error. If you are analyzing a VVDS, you can also focus on particular catalogs, including or excluding VVDS records belonging to certain catalogs.

This is done using the INCLUDE or EXCLUDE parameters:
- INCLUDE—check only those entries meeting the specification
- EXCLUDE—check the complement of the entries meeting the specification.

Although INCLUDE and EXCLUDE define the scope of the catalog analysis, some entries which are not to be checked may need to be scanned during the processing of the command. Errors in scanned records may result in messages, even though the entries were not to be checked.

Table 11 outlines what happens in processing when INCLUDE and EXCLUDE are specified with DIAGNOSE.

*Table 11. DIAGNOSE Processing When INCLUDE or EXCLUDE are Specified*

| Data Set Type | Normal Processing | INCLUDE Processing | EXCLUDE Processing |
|---|---|---|---|
| VSAM cluster | Cluster and all components and paths | VSAM cluster and components but no paths | VSAM cluster and components but no paths |
| VSAM component | Cluster and all components and paths | Only the component | Only the component |
| VSAM path | Path and related alternate index or base cluster | Only the path entry | Only the path entry |
| non-VSAM | Entry and any aliases | Entry only | Entry only |
| non-VSAM alias | Alias and the base entry | Alias entry only | Alias entry only |
| generation data group base | generation data group base, generation data sets, and aliases | generation data group base and generation data sets | generation data group base and generation data sets |
| generation data set | generation data group base, generation data sets, and aliases | generation data set only | generation data set only |

## Processing Considerations for DIAGNOSE

DIAGNOSE checks the content of BCS and VVDS records in the following order:
1. entry or record format
2. any associations (in the BCS only)
3. miscellaneous length and context
4. BCS and VVDS dependencies (if the COMPARE DD option is specified)

DIAGNOSE issues message IDC21364I if errors are discovered in any of these steps. However, once an error is discovered, DIAGNOSE stops processing that entry, and proceeds to the next entry. Thus, additional errors might be hidden.

Because the DIAGNOSE command checks the content of catalog records, if the records contain corrupted information, there is the possibility that the DIAGNOSE job will abend. For example, corrupted length field values could lead DIAGNOSE to attempt to access invalid storage. If DIAGNOSE abends because of corrupted data in the catalog, try to determine the incorrect record, delete it, and recover the data set from a backup copy.

Another point to keep in mind is that DIAGNOSE can indicate that an entry is in error if another job is processing the entry at the same time as DIAGNOSE. For example, if DFSMShsm is migrating a data set while DIAGNOSE is checking the entry for the data set, you might receive an error for the entry. If you suspect that a DIAGNOSE indicated error is erroneous, use DIAGNOSE again and specify the entry in the INCLUDE parameter.

# Analyzing DIAGNOSE Output

If DIAGNOSE finds errors, it issues messages explaining the errors. The record in error is also dumped, unless you specify NODUMP.

The messages provided by DIAGNOSE can result in the following summaries:
* A list of all entries that had no errors
* A list of entries that had errors
* A list of volume serial numbers that were found to be associated with the BCS that were not encountered during a BCS entry scan
* A list of BCS names found to be associated with a given VVDS that were not encountered during a VVDS entry scan
* A list of comparison members that were not encountered during processing
* A list of included or excluded members that were not encountered during processing.

The output of DIAGNOSE generally consists of three error messages. The first, message IDC21364I, provides the following information:
* The name (and type) of data entry being checked
* The key of the record for a BCS or the RBA of the record for a VVDS
* The offset to the start of the cell causing the error
* A reason that describes the error.

The second message is usually IDC21365I, which provides a display of the records in error. If this record was previously displayed, the record is not displayed again. If the record is a DSCB, the first 44 bytes of the record are not displayed.

The third message produced by DIAGNOSE is IDC21363I, which is a summary of all the errors found by DIAGNOSE.

## Overview of DIAGNOSE Messages
DIAGNOSE issues three general types of messages:
* Invocation error messages, which identify errors in command syntax.
* Execution error messages, which identify errors found while analyzing BCS or VVDS records, or VTOC DSCBs. These errors indicate that the input BCS or VVDS is damaged, and the indicated conditions should be corrected to maintain a usable catalog.

- Summary messages, which identify conditions that are possibly, but not necessarily, errors, and summarize the results of the DIAGNOSE job.

Table 12 summarizes the messages which DIAGNOSE issues, and indicates the message type and associated condition code. Complete information concerning most of these messages can be found in *z/OS MVS System Messages, Vol 1 (ABA-AOM)* through *z/OS MVS System Messages, Vol 10 (IXC-IZP)*.

*Table 12. DIAGNOSE Messages*

| Message Number | Condition Code | Message Type |
| --- | --- | --- |
| IDC01360I | 0 | Summary |
| IDC01371I | 0 | Execution |
| IDC01379I | 0 | Execution |
| IDC11361I | 4 | Summary; may be syntax |
| IDC11362I | 4 | Summary |
| IDC11367I | 4 | Summary |
| IDC11373I | 4 | Summary |
| IDC11374I | 4 | Summary |
| IDC11375I | 4 | Summary |
| IDC21363I | 8 | Summary |
| IDC21364I | 8 | Execution |
| IDC21365I | 8 | Execution |
| IDC21372I | 8 | Execution; may be syntax |
| IDC31366I | 12 | Syntax |
| IDC31368I | 12 | Syntax |
| IDC31369I | 12 | Execution |
| IDC31370I | 12 | Execution; may be syntax |
| IDC31376I | 12 | Execution |
| IDC31377I | 12 | Execution |

**Note:**

Condition Code:
**0**       Not an error condition; informational only
**4**       Possible error condition, processing continues
**8**       Error condition, processing continues
**12**       Severe error, processing terminates

# Example of DIAGNOSE Output

Figure 6 shows an example of the output created when comparing a BCS to a VVDS. The records in error are dumped. An error limit of 1 is used, so that only one error is listed.

```
 /**********************************/
 /* DIAGNOSE THE BCS OF DIAGCAT1:  */
 /* -PASS THE BCS NAME ON CATDD    */
 /* -COMPARE THE BCS AND THE VVDS  */
 /*  OF SYS1.VVDS.V339001          */
 /* -DUMP ENTRIES IN ERROR         */
 /* -AFTER ONE ERROR, STOP THE JOB */
 /**********************************/
 DIAGNOSE ICFCATALOG -
          INFILE(CATDD) -
          COMPAREDD(VVDSDD) -
          ERRORLIMIT(1)

IDC21364I ERROR DETECTED BY DIAGNOSE:

  ICFCAT ENTRY:  SAMPLE.KSDS1.DATA (D)

  RECORD:  SAMPLE.KSDS1 /OO

  OFFSET:  X'0073' ◄─────────────────────────── Offset to start of cell
                                                 causing error
  REASON:  12 - CATLG AND VVDS NAMES UNEQUAL

IDC21365I ICFCAT RECORD DISPLAY:

  RECORD:  SAMPLE.KSDS1 /OO

000000  00DB0034  C3004600  2DE2C1D4  D7D3C54B  D2E2C4E2  F1404040  40404040  40404040  *....C....SAMPLE.KSDS1       *
000020  40404040  40404040  40404040  40404040  40404040  40000012  01E5E2C1  D4E3C5E2  *                 ....VSAMTES*
000040  E3408033  7F00000F  0019C400  490012E2  C1D4D7D3  C54BD2E2  C4E2F14B  C4C1E1C1  *T .."....D....SAMPLE.KSDS1.DATA*
000060  00001201  FFFFFFFF  FFFFFFFF  0080337F  00000F00◄IE0400F3  F3F9F0F0  F1305020  *..............".......33900.&.*
000080  09820000  00200000  00000000  00001AC9  004A0013  E2C1D4D7  D3C54BD2  *.................I....SAMPLE.K*
0000A0  E2C4E2F1  4BC9D5C4  C5E70000  1201FFFF  FFFFFFFF  FFFF0080  337F0000  0F001E04  *SDS1.INDEX..............."......*
0000C0  00F3F3F9  F0F0F130  50200982  00000020  00000000  00000000  000000            *.339001.&...................  *

IDC21365I VVDS RECORD DISPLAY:

  RECORD:  X'00002000'

000000  01170040  E9000000  000012E2  C1D4D7D3  C54BD2E2  C4E2F14B  C4C1E3C1  000DE2C1  *... Z......SAMPLE.KSDS1.DATA..SA*
000020  C1C1C1C1  4BD2E2C4  E2F10008  C4C9C1C7  C3C1E3F1  0DE2C1D4  D7D3C54B  D2E2C4E2  *AAAA.KSDS1..DIAGCAT1.SAMPLE.KSDS*
000040  F1000035  21402000  00000600  00000100  00018000  00000000  00280000  00006400  *1....  .....................*
000060  0OFFFFFF  FFFFFFFF  FF000000  00000000  00000000  62608000  60000000            *...........................-..-...*
000080  00004000  00001400  00000000  00000002  00000000  64000000  00000000  00000000  *................................*
0000A0  00000000  00000000  00000000  00000000  01000000  00000000  00000000            *................................*
0000C0  00000000  00000000  00000028  00000000  00000000  00000000  00003E23  80010000  *................................*
000100  00001400  01000000  06000000  06000100  00000000  0027FF                        *.....................    *

IDC31369I MAXIMUM ERROR LIMIT REACHED, PROCESSING TRUNCATED

IDC21363I THE FOLLOWING ENTRIES HAD ERRORS:

  SAMPLE.KSDS1.DATA (D) - REASON CODE:   12

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 8

IDC0002I IDCAMS PROCESSING COMPLETE.  MAXIMUM CONDITION CODE WAS 8
<?Pub *0000003537>
```

DA6C1010

*Figure 6. Sample DIAGNOSE Output*

Message IDC21364I indicates that the entry record for SAMPLE.KSDS1.DATA, the data component of a VSAM cluster, is in error. This record is dumped in message IDC21365I, which also identifies the cluster name, SAMPLE.KSDS1. The corresponding VVDS record is also dumped.

The error detected is "CATLG AND VVDS NAMES UNEQUAL" and the associated reason code is 12. The offset provided points to the beginning of a volume cell. The cell contains the volume serial number 339001, which indicates the VVDS is SYS1.VVDS.V339001. Thus, the entry for SAMPLE.KSDS1.DATA in the BCS does not agree with the entry in SYS1.VVDS.V339001.

An inspection of the VVDS entry shows that the name SAMPLE.KSDS1 appears as SAAAAA.KSDS1. To recover, correct the VVDS record by deleting it and then importing SAMPLE.KSDS1.

Message IDC31369I indicates that the DIAGNOSE job was ended prematurely, because of the error limit.

Finally, message IDC21363I lists all the errors detected in this job before the error limit was reached. This message summarizes information from all IDC21364I messages issued during the job.

# Recovering from Errors Identified by DIAGNOSE

The recovery procedures for DIAGNOSE errors depend on whether the error exists in a BCS entry or in a VVDS entry.

### Recovering Damaged BCS Entries

Use the following steps to recover a damaged BCS entry:

1. Remove the sphere or base record, if it exists.

   The damage detected might not be in a sphere or base record. If it is not, the entry name of the sphere or base record is indicated in messages IDC21364I and IDC21365I.

2. Remove any remaining association records.

   You can re-execute the DIAGNOSE command after you remove the sphere or base record to identify any unwanted truename or association entries in the BCS. You can remove these entries by using the DELETE command with the TRUENAME parameter.

3. Reintroduce the removed entries into the catalog.

   After the damaged entries have been removed, you can redefine the data sets. For VSAM and SMS-managed non-VSAM data sets, you should specify the RECATALOG option of the DEFINE command.

If you are recovering generation data group entries, use the same procedure. However, you must reintroduce the current generation data sets into the catalog in the proper order after the generation data group has been redefined. You can use the LISTCAT command to determine the current generation data sets.

See "Deleting BCS Records" on page 92 and "Recataloging Data Sets and VSAM Objects" on page 91 for more information about correcting BCS entries.

### Recovering Damaged VVDS Entries

Use the following steps to recover a damaged VVDS entry:

1. Remove the entries in the BCS for the data set, if they exist.

   Before the damaged VVDS records can be removed, you must remove the entries in the BCS. See "Deleting BCS Records" on page 92 for more details on removing BCS entries.

2. Remove the damaged VVDS records.

   After you have removed the BCS entries, you can remove the VVDS records by using the DELETE command and specifying VVR or NVR. DELETE VVR or NVR also removes the Format 1 DSCB from the VTOC.

3. Recover the data set from a backup copy.

If a backup copy of the data set does not exist and the data set can be opened, you can attempt to recover some of the data. Depending on the extent and type of

damage in the VVDS record, you might be unable to recover any data. The data that you do recover might be damaged or out of sequence.

See "Deleting VVDS Records and VTOC DSCBs" on page 93 for more information about removing VVDS records.

# Chapter 8. Working with the Catalog Address Space

You can use the MODIFY CATALOG operator command to work with the catalog address space (CAS) and catalogs cached in the in-storage catalog or the catalog data space cache. Using MODIFY CATALOG, you can lessen the impact of catalog recovery, diagnose problems with CAS, and adjust CAS performance.

By using MODIFY CATALOG, you can remove damaged catalog control blocks from CAS without performing a system IPL. You can also remove inaccurate VVDS control blocks after a volume recovery.

You can also use this command to obtain reports on CAS and the catalog data space cache.

## The Catalog Address Space

Catalog functions are performed in the "catalog address space" (CAS). Most catalog modules and control blocks are located in the catalog address space above 16MB. This reduces the required virtual storage in a user's private area needed to perform catalog functions.

During the initialization of a z/OS system, all user catalog names identified in the master catalog, their aliases, and their associated volume serial numbers are placed in tables in CAS. The number of CAS service tasks specified in the SYSCAT*xx* member of SYS1.NUCLEUS (or the LOAD*xx* member of SYS1.PARMLIB) is also created, and a table called the CRT keeps track of these service tasks.

Changes to the master catalog are automatically reflected in the CAS tables. The information in the master catalog is normally the same as the information in CAS. For shared catalogs, the catalog address spaces on all the sharing systems are updated, maintaining data integrity for your systems.

When a user requests a catalog function, a service task is assigned for that request. This task is assigned a CAS ID, which can later be used to end or abend the request if, for any reason, the request is not satisfied.

The catalog address space is not limited by the number of tasks specified in the SYSCAT*xx* member. If the lower limit is set below the CAS maximum limit of 180, CAS allocates new tasks as needed to satisfy user requests up to the CAS limit. These "extra" tasks are then released after they are finished, until the number of tasks reaches the lower limit defined in the SYSCAT*xx* member.

CAS also maintains four special tasks for its own use: the mother task, allocate task, analysis task, and modify task.

The CAS mother task keeps track of all CAS service tasks and other functions of the catalog address space. If the mother task is ended or abended, then all service tasks are ended and CAS is restarted.

The CAS allocate task performs VVDS and catalog allocation as needed. This marks a resource as being used by the task requesting it. If the allocate task allocates a catalog to CAS, the catalog remains allocated until the next system IPL, or until MODIFY CATALOG,UNALLOCATE is used on the catalog.

The CAS analysis task is dedicated to checking CAS for errors. This error checking is performed periodically.

The CAS modify task is used by the MODIFY CATALOG operator command. Only one MODIFY command can be processed at one time. If the modify task is active and another MODIFY CATALOG command is entered, the second command is rejected.

While they are processing, some of the MODIFY CATALOG options require control of specific system resources. If the resource required by the modify task is not available, the task waits a limited time for the resource. If the request is not completed in the allotted time, the CAS modify task abends with abend code A1A. A new modify task is then attached, so that the MODIFY CATALOG command is still available.

The MODIFY command can be entered at any console that can submit operator commands.

## Using MODIFY CATALOG with System Maintenance Procedures

**Operator Commands Discussed in This Section**

MODIFY  CATALOG,CLOSE
MODIFY  CATALOG,SYS%
MODIFY  CATALOG,UNALLOCATE
MODIFY  CATALOG,VCLOSE
MODIFY  CATALOG,VUNALLOCATE

The MODIFY CATALOG command can simplify certain system maintenance procedures. However, MODIFY CATALOG options do not prevent users from using catalog resources. If a maintenance procedure is adversely affected by not having exclusive control of a catalog or VVDS, either do not use MODIFY CATALOG or use it in conjunction with other commands.

## Recovering a Volume Containing a BCS or VVDS

When you recover a volume, vary the volume offline to sharing systems. This is done with the VARY command. However, VARY might not work because a VVDS or BCS on the volume is allocated to the catalog address space. Use MODIFY CATALOG,OPEN to determine if there are any open BCSs on the volume.

You can unallocate a BCS from CAS with the MODIFY CATALOG,UNALLOCATE command. The VVDS can be unallocated with the MODIFY CATALOG,VUNALLOCATE command. Neither of these MODIFY CATALOG commands locks the BCS or VVDS, so issue the VARY command before unallocating them. The DISPLAY command can be used to help determine if the VVDS or BCS on the volume is allocated.

When you recover a volume, the physical location of the volume's VVDS might change. Because the catalog address space maintains control blocks for each VVDS, this change of location might make VSAM and SMS-managed data sets on the volume inaccessible.

If you unallocate the VVDS, the control blocks are rebuilt after a request for the volume is processed by catalog management. If data sets remain inaccessible after the recovery, close the catalog with MODIFY CATALOG,VCLOSE. If data sets remain inaccessible, the problem is not with the control blocks for the VVDS.

The same considerations hold true if there is a BCS on the volume: the control blocks for the BCS are rebuilt by the first request which uses the catalog. If you have problems accessing the catalog, unallocate it or close it. Doing either solves problems of inaccurate control blocks.

Do not use MODIFY CATALOG in place of varying a volume offline. Closing or unallocating a VVDS or a BCS does not prevent users from accessing a volume. MODIFY CATALOG is best used to unallocate a BCS or VVDS from the catalog address space, so that the VARY command can be performed.

## Applying PTFs to the Catalog Component

To activate program temporary fixes (PTFs) and other service applications to the modules that make up the catalog component, do either of the following:

- IPL the system. You must perform an IPL if maintenance changes any load modules other than IGG0CLX0, since those load modules are in SYS1.LPALIB. You must also specify CLPA in response to message IEA101A, or use the MLPA function to temporarily bring in the changed modules.
- Refresh the LINKLIST LOOKASIDE by issuing the MODIFY LLA,REFRESH command. Then perform a restart of the Catalog Address Space. A restart can be done by issuing the MODIFY CATALOG,RESTART operator command, described later in this chapter. This method can be used if the only load module changed by maintenance is IGG0CLX0.

If you change load modules other than IGG0CLX0 and do not follow the procedures that are outlined above, maintenance will not be properly applied to the running system. You might receive unpredictable results, or problems that maintenance should have fixed might still occur.

## Applying PTFs to Systems Using the Storage Management Subsystem

When you are using the Storage Management Subsystem, you cannot use a JOBCAT or STEPCAT DD statement to orient a job to another system's master catalog. In order to apply PTFs (program temporary fixes) or other maintenance to all sharing systems from one system, you can use the SYS% conversion facility. For a complete description of the SYS% facility, see "Using the SYS% Conversion Facility" on page 13.

If the SYS% facility has not been activated, you can use MODIFY CATALOG,SYS%ON to activate it. When you are finished applying the PTFs, you can deactivate the facility with MODIFY CATALOG,SYS%OFF. The current setting of SYS% (on or off) can be determined using MODIFY CATALOG,REPORT.

Before you can use the SYS% facility to orient jobs to system data sets on shared systems, you must define appropriate alias names for each master catalog.

For example, assume you need to apply a fix to SYS1.LINKLIB on three systems: SYSTEMA, SYSTEMB, and SYSTEMC. The SMP/E job runs on SYSTEMA.

Each master catalog must be connected to the other master catalogs as a user catalog, and assigned a four character alias whose first three letters are SYS. The

fourth character can be any valid character. In this case, SYSTEMB has the alias SYSB, and SYSTEMC has the alias SYSC. These aliases are defined in the master catalog for SYSTEMA.

The SMP/E job can then be directed to apply the fix to SYS1.LINKLIB (on SYSTEMA), SYSB.LINKLIB (on SYSTEMB), and SYSC.LINKLIB (on SYSTEMC).

When allocating SYSB.LINKLIB, the system uses the SYSB alias to orient the catalog request to SYSTEMB's master catalog. SYSTEMB's master catalog is then searched for SYSB.LINKLIB. Finding no data set by that name, SYSTEMB's master catalog is again searched for SYS1.LINKLIB, which it finds. The job proceeds using SYSTEMB's SYS1.LINKLIB.

The same procedure is then used to allocate SYSC.LINKLIB.

See "Restrictions on Using SYS% Conversion" on page 15 for restrictions on using the SYS% facility.

# Obtaining Information about Cached Catalogs and CAS

| Operator Commands Discussed in This Section |
| --- |
| MODIFY  CATALOG,ALLOCATED<br>MODIFY  CATALOG,DUMP<br>MODIFY  CATALOG,ENTRY<br>MODIFY  CATALOG,LIST<br>MODIFY  CATALOG,LISTJ<br>MODIFY  CATALOG,ECSHR<br>MODIFY  CATALOG,REPORT<br>MODIFY  CATALOG,REPORT,PERFORMANCE |

You can use the MODIFY CATALOG command to list information about catalogs currently allocated to the catalog address space. Sometimes you need this information so that you can use another MODIFY command to close or otherwise manipulate a catalog in cache.

Table 13 provides an overview of the reporting capabilities of the MODIFY CATALOG command.

*Table 13. Reporting Capabilities of MODIFY CATALOG*

| Option | Message | Purpose |
| --- | --- | --- |
| ALLOCATED | IEC348I | Provides information on all catalogs referred to since the last IPL. |
| DUMP | | Provides dynamic dumping of the catalog address space during diagnostic testing. |
| ENTRY | IEC349I | Provides the storage address and PTF level of a catalog management load module, so that a serviceability level indication processing (SLIP) trap can be set.<br><br>For more information on SLIP traps, refer to *z/OS MVS System Commands*. |

*Table 13. Reporting Capabilities of MODIFY CATALOG  (continued)*

| Option | Message | Purpose |
|---|---|---|
| LIST | IEC347I | Provides the task identification and address of catalog address space tasks, so that the ID can be used in other MODIFY CATALOG commands. The listing also supplies information about job names, elapsed time of the job, and other selected information. |
| LISTJ | IEC3471 | Provides the information about a catalog address space task that is operating on behalf of a user job. |
| ECSHR | IEC3801 | Provides information about the current status of the enhanced catalog sharing (ECS) function, and catalogs that might be using the facility. |
| REPORT | IEC359I | Provides general information on the status of the catalog address space. Current settings for number of tasks, alias search level, CRT slot rotation, and SYS% conversion are listed, as well as the addresses of the CAS mother, modify, analysis, and allocation tasks.<br><br>There are other forms of the REPORT command that provide information on other aspects of the catalog address space. |
| REPORT, PERFORMANCE | IEC359I | Displays information about the performance of specific events that catalog processing invokes. Each line shows the number of times (*nnn*) that event has occurred since IPL or the last reset of the statistics by MODIFYCATALOG,REPORT,PERFORMANCE(RESET), and the average time for each occurrence (*nnn.nnn*). The unit of measure of the average time (*unit*) is either milliseconds (MSEC), seconds (SEC), or the average shown as hours, minutes, and seconds (*hh:mm:ss.th*).<br><br>Other forms of the REPORT command provide information on different aspects of the catalog address space. |

**Note:** This table does not reflect the detailed syntax of these commands. See "MODIFY CATALOG Command Syntax" on page 125 for syntax information.

## Monitoring the Catalog Address Space

Using the MODIFY CATALOG,REPORT command, you can obtain general information about the catalog address space. This information can be used to evaluate your current catalog environment setup. If you determine your current setup is inadequate, you can change it with another MODIFY CATALOG command, or by changing the SYSCAT*xx* member of SYS1.NUCLEUS.

The following example shows the output of an unqualified MODIFY CATALOG,REPORT command:

```
*CAS**************************************
*   CATALOG COMPONENT LEVEL   = HDZ11F0   *
*   CATALOG ADDRESS SPACE ASN = 0022      *
*   SERVICE TASK UPPER LIMIT  =   180     *
*   SERVICE TASK LOWER LIMIT  =   200     *
*   HIGHEST # SERVICE TASKS   =     5     *
*   CURRENT # SERVICE TASKS   =     5     *
*   MAXIMUM # OPEN CATALOGS   = 1,024     *
*   ALIAS TABLE AVAILABLE     = YES       *
*   ALIAS LEVELS SPECIFIED    = 1         *
*   SYS% TO SYS1 CONVERSION   = OFF       *
*   CAS MOTHER TASK           = 008F49A8  *
*   CAS MODIFY TASK           = 008F4788  *
*   CAS ANALYSIS TASK         = 008F4138  *
*   CAS ALLOCATION TASK       = 008F4460  *
*   VOLCAT HI-LEVEL QUALIFIER = SYS1      *
*   DELETE UCAT/VVDS WARNING  = ON        *
*   DATA SET SYNTAX CHECKING  = xxxxxxxx  *
*CAS**************************************
```

The first two entries supply information on the component level of catalog management, and the ASID for the catalog address space.

The next four entries supply information about the service tasks available to process catalog requests. In this example, you can see that the **service task lower limit** is adequate for the **current number of tasks**. The **service task upper limit** defaults to 180, and can be changed by the MODIFY CATALOG,TASKMAX command. The **highest number of service tasks** is equal to the highest value the current number of tasks field has reached.

The last entry for the data set syntax checking is set to either ENABLED or DISABLED.

Catalog management creates tasks as necessary as the current number of tasks exceeds the service task lower limit. As catalog requests subside, the number of tasks attached and available for processing requests is reduced until the lower limit is reached.

**Maximum number of open catalogs** is the value set by the MODIFY CATALOG,CATMAX command. This is the maximum number of catalogs which might be open to CAS simultaneously. If the maximum is reached, catalog management closes the least recently used catalog before opening another catalog. Normally, a catalog remains open once it has been opened.

**Alias table available** entry indicates whether there is a problem with the catalog alias table. This field should always say YES. If it says NO, try restarting the catalog address space. Performance is affected if catalog management does not have catalog aliases in the catalog alias table.

**Hint:** The alias table might not have been created if there was not enough virtual storage available during system initialization. Other messages are issued to indicate this problem. If restarting CAS fails to create the alias table, determine why there is not enough virtual storage, and make the necessary changes to your system.

**Alias levels specified** is the search level for the multilevel alias facility. It is set either at system initialization by the SYSCAT*xx* member of SYS1.NUCLEUS, or by the MODIFY CATALOG,ALIASLEVEL command.

**SYS% to SYS1 conversion** entry indicates whether the SYS% facility is on or off.

The next four entries indicate the hexadecimal addresses of the indicated tasks.

**VOLCAT hi-level qualifier** entry indicates the high level qualifier used for the tape volume catalogs of this system.

**Delete UCAT/VVDS warning** indicates whether write-to-operator with reply (WTOR) messages, IDC1997I and IDC1998I, will be issued for an IDCAMS DELETE UCAT FORCE or DELETE VVDS FORCE (ON), or if they will not be issued (OFF). The setting is changed by the MODIFY CATALOG WARNING or MODIFY CATLOG NOWARNING commands.

**Data Set Syntax Checking** indicates whether or not the validity of the data set name is checked prior to adding a record to the catalog. The name must conform to the syntax rules defined in *JCL Reference*. If you have programs that create cataloged names that do not meet the syntax requirements you may disable this checking. However, this may result in the creation of catalog entries that cannot be removed by IDCAMS or any other IBM utilities.

## Monitoring the Catalog Address Space Performance

The MODIFY CATALOG,REPORT,PERFORMANCE command can be used to examine certain events that occur in the catalog address space. These events represent points at which catalog code calls some function outside of the catalog component, such as enqueues, I/O, or allocation. All such events are tracked, except for z/OS lock manager requests and GETMAIN/FREEMAIN activity. An example of the output from this command is:

```
IEC359I CATALOG PERFORMANCE REPORT
*CAS***************************************************
*  Statistics since 12:14:49.61 on  09/29/2000       *
*  -----CATALOG EVENT----   --COUNT--  ---AVERAGE---  *
*  Entries to Catalog          313     23.418  SEC   *
*  BCS ENQ Shr                 291      0.084 MSEC    *
*  BCS ENQ Excl                 20      0.082 MSEC    *
*  BCS DEQ                     352      0.082 MSEC    *
*  VVDS RESERVE CI              14      0.077 MSEC    *
*  VVDS DEQ CI                  14      0.176 MSEC    *
*  VVDS RESERVE Shr             61      0.153 MSEC    *
*  VVDS RESERVE Excl             2      0.099 MSEC    *
*  VVDS DEQ                     63      0.147 MSEC    *
*  SPHERE ENQ Excl               2      0.054 MSEC    *
*  SPHERE DEQ                    2      0.033 MSEC    *
*  RPL ENQ                       3      0.210 MSEC    *
*  RPL DEQ                       3      0.244 MSEC    *
*  BCS Get                   2,181      1.102 MSEC    *
*  BCS Put                       2     12.836 MSEC    *
*  BCS Erase                     3     34.630 MSEC    *
*  VVDS I/O                     81      9.310 MSEC    *
*  VLF Define Major              1      3.879 MSEC    *
*  VLF Identify              4,596      0.000 MSEC    *
*  RMM Exit                      8      0.005 MSEC    *
*  Tape Exit                     8      0.001 MSEC    *
*  BCS Allocate                  3     13.102 MSEC    *
*  SMF Write                     9      0.197 MSEC    *
*  VVDS Format                   5     23.878 MSEC    *
*  IXLCONN                       2      1.511 MSEC    *
```

```
        *  MVS Allocate                   2    267.105 MSEC  *
        *  Lookup/Pin UCB                 8      0.329 MSEC  *
        *  Unpin UCB                      8      0.340 MSEC  *
        *  Capture UCB                    3      0.062 MSEC  *
        *  Uncapture UCB                  6      0.061 MSEC  *
        *  RACROUTE Auth                 47      5.511 MSEC  *
        *  RACROUTE Define                3    537.213 MSEC  *
        *  DADSM Allocate SMS             5     15.163 SEC   *
        *CAS**************************************************
```

This example does not show all of the possible events that may appear as output from this command.

This command can be useful in identifying performance problems that you suspect are related to catalog processing. For example, if the average time for ENQS that is shown in the report seems excessive, it might indicate some problems in the RS configuration, or parameter specifications. High I/O times might indicate problems:

- With channel or device load
- With volumes that are suffering a high number of I/O errors
- With volumes that have excessively high RESERVE rates or long RESERVE durations.

In the previous example, the last entry identifies an average of over 15 seconds per DADSM allocate request. This might indicate some problem with reserved volumes, or I/O errors.

To reset the information in this report, you can issue the MODIFY CATALOG,REPORT,PERFORMANCE(RESET) command.

## Evaluating Catalog Data Space Cache Performance

In order to evaluate the catalog data space cache, use the MODIFY CATALOG,REPORT,CACHE command. You can use the command to evaluate the cache performance for a specified catalog, or for all catalogs that are currently being cached. The numbers shown in this report are in decimal. If a 'K' appears after the number, it means that the number has been divided by 1024. Following is an example of the output for a catalog cache statistic report:

```
IEC359I CATALOG CACHE REPORT
*CAS***************************************************************
*   HIT% -RECORDS- -SEARCHES --FOUND-- -DELETES- -SHR UPD- --PURGE-- *
*                                                                  *
* SYS1.MVSRES.MASTCAT                      (ISC)                    *
*   42%   1,578       679       291       13          0        3 *
* SYS1.PROD.UCAT                           (VLF)                    *
*   37%   2,304     1,014       372       27         42        0 *
*                                                                  *
*CAS***************************************************************
```

The report on SYS1.MVSRES.MASTCAT, currently using ISC caching, shows that 42% of the records requested from this catalog were already in storage. Therefore, no I/O operation was necessary. SYS1.PROD.UCAT is currently being cached through the VLF function, and shows similar statistics. Twenty percent is the break-even point for catalog caching. If the hit percentage is lower than 20, then the processor storage and cycles needed to maintain the data space are not worth the I/O operations saved. This figure is a rough estimate. If central processing unit (CPU) utilization and processor storage are not limiting factors, a lower hit ratio is tolerated.

When you evaluate the cache performance for a catalog, you need to consider how long the catalog has been using the cache. If the cache has only been available for a catalog for an hour, the hit ratio will likely be low. However, if the catalog has been using the cache for a week, expect a good hit ratio. The hit ratio is an indication of cache usage while the cache is available. The values do not accumulate between performing IPLs, stopping and starting VLF, or restarting the catalog address space.

To obtain a good comparative analysis of how different catalogs are using the cache, produce periodic reports during an active session (between IPLs). This shows you which catalogs are getting good hit ratios quickly, and those which are either reaching the break-even point slowly or not at all. If you want to reset the statistics for a catalog, you can use the MODIFY CATALOG,CLOSE command to close that catalog. When it is next referenced, a new cache structure will be built.

To calculate the hit ratio, divide the number of hits by the number of searches of the data space. The Catalog Cache Report example also lists the number of catalog records in the cache and the number of deleted or updated records in the cache. Note that for catalogs using VLF, the count of records in the cache might not be accurate due to VLF trimming. The number that is shown represents the number of records that the catalog has added to VLF, not the number that might actually be in the VLF cache.

Hit ratio results are relative to a particular installation, workload, or reference pattern. It is possible (but not guaranteed) to increase hit ratios by dedicating more storage space to a VLF-managed catalog. For more information on how to dedicate more storage space to a VLF-managed catalog, refer to *z/OS MVS Initialization and Tuning Reference*.

The master catalog will always be searched for data set names that do not have an alias to a user catalog. For example, non-existent data set names and SMS-managed temporary data sets are not cataloged so all searches for them would show up as a ″miss″ of the master catalog cache. This may cause the hit ratio for the master catalog cache to appear lower than expected.

The PURGE count displays the number of times a particular catalog cache has been purged and is reset at an IPL (Initial Program Load). Catalog purges commonly occur for two reasons:

- ICS-cached catalog purges occur when a change is made from an external system and is recognized by the current system.
- The number of changes within a system exceeds the maximum that can be recorded between two catalog access events. The specific number of changes that can be recorded is an internal value and varies based on several conditions, all of which are outside of the user's control.

Catalog management allows for record level granularity updates to the VLF cache in a cross system sharing environment. Record updates for a catalog using ISC caching cause the entire ISC cache to be purged when a change is made from another system. Nonzero values under the PURGE column for a catalog using ISC caching indicate this is happening for that catalog. Deletes or updates to a shared catalog on one system cause a corresponding deletion of records in the VLF cache on the other systems. Catalog management accomplishes this by using a list of entries in the catalog that describes VVR.

Records might not be found in the cache of the corresponding system because they were never accessed on that system. The cache is a caching system where records are not added to the cache until they are accessed. The catalog caching report shows a value for sharing (SHR UPD). This sharing number reflects how many records that were found and deleted from the VLF cache because they were updated or deleted on a shared system. The deleted number given in the catalog data space report reflects:

- The *total* count of records updated or deleted on the current system AND
- Those records updated or deleted on the shared system which are found in the current system's cache.

If more updates or deletions occur in a given time than can be processed by the sharing system, the entire VLF cache on the sharing system is invalidated. This is shown in the MODIFY CATALOG,REPORT,VLF output under the heading PURGE. This number should always be small, otherwise this indicates that more updates are occurring than can be processed by the sharing system. If the purged number grows dramatically over a short period of time, the catalog is probably not a good candidate for caching.

The numbers shown for record counts, records, searches, hits, deletes, and sharing might not have a one-to-one correlation to the catalog functions being performed. This is because catalog management maintains other records, such as truename records, that can be cached and will add to the statistics.

If a catalog is not using the catalog data space cache to your satisfaction, you can temporarily make it ineligible for CDSC by using MODIFY CATALOG,NOVLF. Alternatively, after closing or deallocating the catalog, you can use the ALLOCATE,NOVLF subparameter to allocate the catalog to CAS without the use of VLF.

To make the catalog permanently ineligible for future data space cache use, modify the COFVLF*xx* member of SYS1.PARMLIB, and stop and restart VLF.

## Obtaining Task Identifiers Needed by Other MODIFY Commands

Some MODIFY CATALOG commands require that you supply a task address or ID, or a volume serial number or catalog name, in order to perform the desired task. The OPEN and LIST parameters can be used to obtain these task identifiers.

For MODIFY CATALOG commands that require a task ID or address (ABEND, END), use the LIST parameter. For commands that require a volume serial number, use OPEN. For commands that require a catalog name, you can use either LIST or OPEN.

The main difference between LIST and OPEN is that LIST provides information about a specific task using the task's job name, ID, or storage address. OPEN lists information about all catalogs allocated, even those not assigned to the catalog address space. OPEN also tells you the type of cache to which a catalog is assigned.

**Restriction:** Catalogs are only assigned or allocated to the catalog address space if they are opened by a standard catalog search, or if they are allocated to CAS with MODIFY CATALOG,ALLOCATE. If the catalog is only opened by a JOBCAT or STEPCAT DD statement (a

non-standard search), the catalog is not assigned to CAS. After the job is completed, JOBCAT and STEPCAT catalogs are closed, and their CAS storage freed.

## Interpreting MODIFY CATALOG,LIST Output

The following is an example of the output for MODIFY CATALOG,LIST:

```
IEC347I LIST CATALOG TASK(S)
*CAS*************************************************************
* FLAGS -  TASK ADDRESS - JOBNAME  / STEPNAME - ELAPSED TIME - ID  *
* -W---L    005AB2A8      ACCTING / SORTSTEP     00.08.26    04 *
*****************************************************************
* O-OLDEST, W-WAIT, A-ABEND, E-ENQ, R-RECALL, L-RLS                *
*CAS*************************************************************
```

The job ACCNTING has been waiting for more than 8 minutes. This is excessive time for a catalog request, and indicates a problem. The ″L″ indicator shows that this request is currently waiting on a response from the RLS address space for some requested function.

You can also use the MODIFY CATALOG,LISTJ(ACCTING),DETAIL command to obtain a more detailed set of information about this particular catalog request. This information is useful to determine if it is waiting on a particular catalog that is unavailable, or an ENQ resource on which there is contention. The information from the LISTJ DETAIL command is useful when reporting a suspected problem to IBM service. Using either the END or ABEND parameters and the task ID or address, you can stop the request. See "Ending a Catalog Request Task" on page 118 for more information about ending CAS tasks.

The flags have the following meaning:

**O**      The task is the oldest active task. This is only indicated if all active CAS tasks are listed.

**W**      The task is waiting for the completion of some event, for example, an ENQ or a tape mount.

**A**      The task is abending.

**E**      The task is waiting for an ENQ on a catalog resource.

**R**      The task is suspended while the requester's address space is recalled to perform a needed function or obtain information. For example, mounting a volume, verifying security, renaming or erasing data, or extending a data set can cause a recall. If the recall lasts for a very long time, examine the system log for messages indicating the delay. This recall has no relationship to any DFSMShsm function.

**L**      The task is suspended waiting for a response from the RLS address space for a request for an RLS function. When this flag displays, you should follow the RLS-specific diagnostic procedures for gathering information to report this problem.

The TASK ADDRESS field gives the hexadecimal address of the CAS service task.

The JOBNAME / STEPNAME field gives the name of the job and step which initiated the catalog request.

The ELAPSED TIME field gives the "hours.minutes.seconds" that the task has been active in CAS.

The ID field gives the CAS identifier for the task.

## Interpreting MODIFY CATALOG,LISTJ(jobname),DETAIL Output

The following is an example of the output for MODIFY CATALOG,LISTJ(jobname),DETAIL command:

```
IEC347I LIST CATALOG TASK(S)
*CAS******************************************************************
* JOB/STEP Name: IBMUSER /IEFPROC     ASN: 0034  TCB: 008C57D0       *
* CAS TCB: 00897D08 Task Number: 01 TCB Comp Code: 00000000          *
* CCX: 068E9000  CCA: 7F73A000  CCAPROB: 00000000                    *
* CTGPL: 0622A103 7F48FD2A 7F48FCCE 7F48FDE0                         *
*        0400FF04 7F48FCFC 00000000                                  *
* Request Type: GFL                                                  *
*   CTGENT:                                                          *
*   CTGCAT:                                                          *
* CCASRCH: SYS1                                                      *
* Oriented to: SYS1.MVSRES.MASTCAT                                   *
* Waiting for completion of: BCS Read                                *
*        at 00B8734C for 00.00.01                                    *
*CAS******************************************************************
```

This command is primarily designed to provide detailed information about a particular catalog request. Use this command to obtain more information about a request that showed excessive processing time after the MODIFY CATALOG,LIST command was issued. Much of this information is designed for IBM Service personnel, but there are some fields that can help you do real-time problem diagnosis:

- CCASRCH - the last name of an entry that is attempting to be accessed (or was accessed) in the catalog the request is oriented to.
- Oriented to - shows the catalog that is being accessed for the indicated request
- Waiting for completion of - indicates a specific event outside of the catalog code that the request is waiting for. This includes the address where catalog code will continue executing when the request is complete, and the length of time the request has been waiting

This information might allow you to perform other real-time diagnosis depending on what the request is waiting on, and what catalog or data set is indicated. For example, if it shows it is waiting on the completion of a BCS ENQ, you might try issuing a D GRS,C command to see if there is contention on that catalog name, and what job in the system might be causing that contention.

There is other information that might or might not be displayed in response to this command, depending on the type and state of a request. The above example does not show all of this information.

## Interpreting MODIFY CATALOG,ALLOCATED Output

The following is an example of the output for MODIFY CATALOG,ALLOCATED:

```
**CAS************************************************************
** FLAGS-VOLSER-USER-CATALOG NAME                               *
****************************************************************
** YSI-R JESRS1 0001 UCAT.VJESRS1                               *
** YSV-- ESARS1 0001 UCAT.VESARS1                               *
** YSV-- ESADL1 0001 UCAT.VESADL1                               *
** YSV-R JESDL1 0001 UCAT.VJESDL1                               *
** YSV-- MVSTS2 0001 UCAT.VMVSTS2.TEST                          *
** YSV-- MVSTS2 0001 UCAT.VMVSTS2                               *
** Y-I-R ESACAT 0001 MCAT.VESACAT                               *
** Y--A- ATLCAT 0001 SYS1.VOLCAT.VGENERAL                       *
****************************************************************
```

```
** Y/N-ALLOCATED TO CAS, S-SMS, V-VLF, I-ISC, C-CLOSED, D-DELETED,*
** R-SHARED A-ATL E-ECS K-LOCKED                                  *
**CAS**********************************************************
```

In this example, all catalogs except the master catalog reside on SMS-managed volumes. All but two use the catalog data space (VLF) cache.

The flags have the following meaning:

**Y/N**   The catalog is (Y) or is not (N) allocated to the catalog address space.
**S**     The catalog is managed by the Storage Management Subsystem.
**V**     The catalog is using the catalog data space cache (VLF).
**I**     The catalog is using the in-storage catalog.
**C**     The catalog is closed.
**D**     The catalog has been deleted.
**R**     The catalog is using cross systems sharing.
**A**     The catalog is a tape volume catalog.
**E**     The catalog is shared and is using the Enhanced Catalog Sharing facility.
**K**     The catalog has been locked by an IDCAMS ALTER LOCK or IMPORT LOCK command.

The VOLSER field shows the volume serial number of the volume containing the catalog.

The USER field contains a count of the number of address spaces that have allocated this catalog. This Catalog Address Space number is normally one. If it is higher than one, it might mean that other jobs are using the catalog as a JOBCAT or STEPCAT.

## Interpreting MODIFY CATALOG,REPORT,DUMP Output

The following is an example of the output for MODIFY CATALOG,REPORT,DUMPON:

```
*CAS**********************************************************
*    STATUS  RETURN CODE  REASON CODE  MODULE ID  COUNT          *
*     aaa        bbb          ccc          dd       eee           *
*CAS**********************************************************
```

The variables have the following values:

**aaa**   The dump status which is either ON or OFF.
**bbb**   The catalog return code in decimal. The range is 0 to 255, or '***'.
**ccc**   The catalog reason code in decimal. The range is 0 to 255, or '***'.
**dd**    The catalog module identifier. or '**'.
**eee**   The match count in decimal. The range is 0 to 999.

This number decrements each time an error that matches the return code, reason code, and module identifier is detected within the catalog address space. When the count becomes zero, a dump will be taken for that occurrence of the error.

The MODIFY CATALOG,DUMPON command allows the specification of asterisks for any two of the fields return code, reason code, or module identifier. The asterisks indicate that field should not participate in a match for a detected error. For example, DUMPON(132,***,FO) will match on any return code 132 issued from IGG0CLFO.

## Interpreting MODIFY CATALOG,ECSHR(STATUS) Output

The following is an example of the output for MODIFY CATALOG,ECSHR(STATUS):

```
*CAS*************************************************************
*  CF Connection: AutoAdd                                       *
* ------------------CATALOG------------------- -----STATUS------ *
* SYS1.VOLCAT.VGENERAL                         Inact(NotShrable) *
* SYS1.MVSRES.MASTCAT                          Active           *
* UCAT1.USERCAT                               Active           *
* UCAT2.USERCAT                               Inact(Disconnect) *
* UCAT3.USERCAT                               Inact(NonECSAcc)  *
*CAS*************************************************************
```

The above example shows that the catalog connection to the coupling facility structure that is used for Enhanced Catalog Sharing is active. If the connection did not exist, return and reason codes that are associated with the failure to connect would be displayed.

The information listed for the status of the ECS facility includes error information if the connection has failed. The possible values for the connection status are:

**Autoadd**
The system is connected to the ECS structure, and the automatic add function is enabled.

**Connect Failure**
While attempting to connect to the ECS structure, an error was returned from IXLCONN. The return and reason codes from IXLCONN are also displayed.

**Connected**
The system is connected to the ECS structure in the coupling facility; AUTOADD is disabled.

**Inact(CFFail)**
The system has disconnected because a coupling facility failure was detected.

**Inact(Disconnect)**
ECS is disconnected from the ECS structure in the coupling facility due to a MODIFY CATALOG command with the ECSHR(DISCONNECT) parameter.

**Inact(Restart)**
The system disconnected while processing a CAS restart.

**Quiescing**
ECS is currently active, but is in the process of disconnecting.

**Rebuild Connected**
ECS has connected to the new structure during rebuild processing.

**Rebuild Cleanup**
ECS has received and processed the rebuild cleanup event.

**Rebuild Quiesced**
ECS activity has been quiesced due to a rebuild request.

**Rebuild Stopped**
A rebuild of the ECS structure that was in progress has been stopped.

**Unknown**
The system status of ECS facility is unknown.

If the connection has been terminated because of a service error, a return code, a reason code, and the module ID will be shown under the system connection status line. The return code and reason code of a coupling facility service call will also be displayed if available.

After the system status is displayed, the ECS status of all catalogs that have been referenced since the last system IPL is displayed. The values of status that might appear are:

**Active**
ECS is active for the catalog.

| | |
|---|---|
| **Inact(CFFail)** | The catalog is inactive because the coupling facility failed. |
| **Inact(CFFull)** | An attempt was made to activate the catalog, but the ECS structure in the coupling facility is full. |
| **Inact(Disconnect)** | The catalog is inactive because the system disconnected from the ECS structure. |
| **Inact(MaxCats)** | An attempt was made to activate the catalog, but the maximum number of catalogs that are allowed in the ECS structure has been reached. |
| **Inact(NeverConn)** | The catalog is inactive because the system is not connected to the ECS structure in the coupling facility. |
| **Inact(NonECSAcc)** | The catalog is inactive because the last system to access the catalog was non-ECS 1.5. system. |
| **Inact(NotElig)** | The catalog is inactive because it is not ECS-eligible; it does not have the ECSHARING attribute. |
| **Inact(NotShrable)** | The catalog is inactive because it either is not on a shared volume or does not have SHAREOPTIONS(3,4). |
| **Inact(Removed)** | The catalog is inactive due to a MODIFY CATALOG command with the ECSHR(REMOVE,...) parameter. |
| **Inact(Unknown)** | The ECS status of this catalog has never been set or is otherwise unknown. |

The status of a catalog is the last status known to the ECS facility. If the status changes, it will not be reflected in this display until the next catalog request that uses that catalog.

## Fixing Temporary Catalog Problems

| **Operator Commands Discussed in This Section** |
|---|
| MODIFY  CATALOG,ABEND |
| MODIFY  CATALOG,CLOSE |
| MODIFY  CATALOG,END |
| MODIFY  CATALOG,RESTART |
| MODIFY  CATALOG,UNALLOCATE |
| MODIFY  CATALOG,VCLOSE |
| MODIFY  CATALOG,VUNALLOCATE |

Occasionally, the control blocks for a catalog kept in the catalog address space might be damaged. You might think the catalog is damaged and in need of recovery, when only the control blocks need to be rebuilt. If the catalog appears damaged, try rebuilding the control blocks first. If the problem persists, recover the catalog.

There might also be situations where a job gets an enqueue lockout from a catalog, or a catalog request is not being satisfied for some reason. In these cases, the job needs to be ended and redriven, if possible.

It is also possible that a catalog request that is currently being processed, cannot be properly retried after being interrupted by these commands. Use these commands when all other means of correcting an ongoing catalog error have failed.

Table 14 gives an overview of the error recovery capabilities of the MODIFY CATALOG command.

*Table 14. Error Recovery Capabilities of MODIFY CATALOGS*

| Option | Purpose |
|---|---|
| ABEND | To end a CAS task abnormally. This should only be used after you have unsuccessfully tried END, or when you are ending the CAS allocate, analysis, or modify tasks. |
| CLOSE | To release all CAS storage for the specified catalog. The catalog is not locked, and is re-opened to CAS by the next catalog request that accesses the catalog. The catalog is not unallocated from CAS, and the common services area storage used by the catalog is not freed. |
| END | To end a CAS task. You can choose to redrive the request, or simply end it. This is the preferred method of ending a CAS task, and should be used before attempting ABEND. |
| RESTART | To abnormally end the CAS mother task and restart it in a new address space. This option should only be used when your only other option is to IPL the system. If RESTART fails, you must IPL. |
| UNALLOCATE | To unallocate and close a catalog from CAS without releasing the CAS storage. As with CLOSE, the control blocks for the catalog are rebuilt with the next request for the catalog, and the catalog is not locked.<br><br>This command is useful for getting a volume offline. |
| VCLOSE | To close the VVDS which resides on the specified volume. The VVDS is opened by the next request which tries to access it. |
| VUNALLOCATE | To unallocate all VVDSs from the catalog address space. This can help in getting a volume offline for recovery. The VVDS is not locked, and the next request which tries to access it opens the VVDS. However, VVDSs are not allocated to the catalog address space until MODIFY CATALOG,NOVUNALLOCATE is issued. |

**Note:** This table does not reflect the detailed syntax of these commands. See "MODIFY CATALOG Command Syntax" on page 125 for syntax information.

## Ending a Catalog Request Task

Using the LIST parameter, you can determine that a catalog request task is taking too much time to execute. If your resource monitor program indicates you have an enqueue lockout on a resource held by CAS, the listing gives you the ID of the task with the enqueue. You might want to use the MODIFY command with the LISTJ and DETAIL keywords to get more information about the associated catalog request.

In these situations, you will want to end the task. Sometimes redriving the task allows the request to be successfully satisfied, especially if there is an enqueue lockout. The timing of the resource requests on the redrive are probably different than when the enqueue occurred, so that the requests are satisfied.

Use the following procedure to end a catalog request task:

1. Use LIST to obtain the task ID or address.
2. Skip this step if you are trying to end the CAS allocate, analysis, or modify task. Otherwise, if you are trying to end a user task,
   a. Use END with REDRIVE if the task is in enqueue lockout, or you want the request to be redriven an unlimited number of times, or
   b. Use END with NOREDRIVE if you want to permanently end the task.

3. If the preceding step failed, or you skipped it, use ABEND to end the task and redrive the request once.

# Refreshing a Catalog's Control Blocks

The catalog address space is designed with internal checks that allow it to identify, and subsequently rebuild, damaged catalog control blocks. However, not all problems can be identified.

If you find that attempts to access a particular catalog are resulting in recurrent abends, rebuild the control blocks in the catalog address space. This should be done using the CLOSE or UNALLOCATE parameters. If the damaged control blocks are for a VVDS, use VCLOSE or VUNALLOCATE.

Each of these parameters causes the control blocks to be released and rebuilt on a subsequent request that tries to access the catalog. Which parameter you use depends on whether you want the catalog unallocated and closed (UNALLOCATE), or closed but left allocated (CLOSE).

When rebuilding the control blocks for a VVDS, use VCLOSE whenever possible. With VCLOSE, you specify the particular VVDS whose control blocks you suspect are in error. VUNALLOCATE can also be used, but this parameter unallocates all VVDSs. You cannot unallocate a specific VVDS.

The UNALLOCATE parameter can be used to unallocate all catalogs at once. It can also be used to unallocate a catalog on a volume which you need to vary offline. Use VUNALLOCATE if you need to unallocate a VVDS to get a volume offline.

# Restarting the Catalog Address Space

The catalog address space is designed to restart with a minimum of interruption to your system. Requests that are being processed are generally able to be restarted from the beginning. Requests that are made while the address space is in the process of restarting are temporarily suspended.

The catalog address space is critical to the functioning of your system, and there is always the possibility that a restart might fail. However, the use of the restart facility might also prevent an IPL or clear other error conditions that are a result of problems associated with the use of catalogs. For example, the following problems are typically corrected by a restart:

- Inability to vary a volume containing a catalog offline, when a MODIFY CATALOG,CLOSE command does not release the volume
- ABENDs in the catalog address space that relates to lack of storage (such as 878 ABENDs)
- ABENDs in the catalog address space that might indicate damage to control blocks (such as repeated 0C4 ABENDs at the same location)
- ENQ lockouts, particularly on the SYSIGGV2 resource, when the MODIFY CATALOG,ABEND command will not remove the task in error
- Installing catalog maintenance to correct a problem when an IPL is not necessary or feasible

Do not use RESTART to refresh catalog or VVDS control blocks or to change catalog characteristics. The use of other MODIFY command formats are designed to accomplish this on a catalog-by-catalog basis. There is a risk that the catalog

address space restart might fail for some unanticipated reason. If this occurs, it will be necessary to IPL the system to recover the address space. However, a restart failure is a very unlikely occurrence.

It is also possible that a catalog request that is currently being processed cannot be properly retried after being interrupted by these commands. For example, this could occur with a DEFINE command of a VSAM data set that is partially completed at the time of the restart. It is recommended that you try to quiesce system activity as much as feasible before doing a restart, or at least attempt to minimize the use of catalogs.

When you issue MODIFY CATALOG,RESTART, the CAS mother task is abended with abend code 81A, and any catalog requests in process at the time are redriven.

The restart of CAS in a new address space should be transparent to all users. However, even when all requests are redriven successfully and receive a return code 0, the system might produce indicative dumps on the console, the system log, and on user job logs. There is no way to suppress these indicative dumps.

As noted above, a request might not successfully be redriven. If this is the case, the appropriate return and reason code information, associated messages, and possibly system dumps will be produced.

The catalog address space is designed to recover from cross-memory failures that can occur during CAS restart. CAS recognizes and recovers from the following abend codes, which might occur during a restart: 052, 058, 066, 070, 073, and 0D$x$. You can ignore any indicative dumps produced by the system for these abend codes. Only the final catalog return code, which should be 0, is significant.

## Making Temporary Modifications to the Catalog Environment

| Operator Commands Discussed in This Section |
| --- |
| MODIFY CATALOG,ALIASLEVEL<br>MODIFY CATALOG,ALLOCATE<br>MODIFY CATALOG,CATMAX<br>MODIFY CATALOG,CLOSE<br>MODIFY CATALOG,{ISC\|NOISC}<br>MODIFY CATALOG,{ROTATE\|NOROTATE}<br>MODIFY CATALOG,{SYS%ON\|SYS%OFF}<br>MODIFY CATALOG,TASKMAX<br>MODIFY CATALOG,UNALLOCATE<br>MODIFY CATALOG,{VLF\|NOVLF}<br>MODIFY CATALOG,{WARNING\|NOWARNING}<br>MODIFY CATALOG,ENABLE(DSNCHECK)}<br>MODIFY CATALOG,DISABLE(DSNCHECK)} |

The MODIFY CATALOG command allows you to alter a number of attributes that are initialized at IPL time. Many of these attributes are initialized by the SYSCAT$xx$ member of SYS1.NUCLEUS. The maximum catalog value (CATMAX) is also set at IPL time to 50 (X'32').

The parameters discussed in this section allow you to tailor your system according to temporary needs, without requiring you to bring down your system and IPL using

a different SYSCAT*xx* member. Other parameters allow you to gain closer control over storage, so that you can free up storage used by catalog management for your own jobs.

Table 15 gives an overview of the temporary system tailoring capabilities of the MODIFY CATALOG command.

*Table 15. Temporary System Tailoring Capabilities of MODIFY CATALOG*

| Option | Purpose |
|---|---|
| ALIASLEVEL | To change the number of qualifiers used to determine the alias of a user catalog. This is the multilevel alias search level. |
| ALLOCATE | To allocate a catalog to the catalog address space. Using the NOISC or NOVLF subparameters, you can prevent the allocated catalog from using the specified cache. |
| CATMAX | To close all open catalogs and set a maximum to the number of catalogs that might be open in CAS. When the maximum is reached, the least recently used catalog is closed. This conserves storage. |
| CLOSE | To close a catalog and remove it from CAS, and to free up the CDSC or ISC storage used by the catalog. |
| DISABLE | To disable specific optional features. See the discussion of the DISABLE keyword for features that can be specified. |
| ENABLE | To enable specific optional features. See the discussion of the ENABLE keywords for features that can be specified. |
| ISC and NOISC | To either place a catalog in the in-storage catalog cache, or remove it from that cache. |
| ROTATE and NOROTATE | To either activate or deactivate rotation through the CRT table. If ROTATE is specified, control blocks will more likely be available for diagnostic purposes. |
| SYS%ON and SYS%OFF | To either activate or deactivate the SYS% facility, to convert searches for SYS% data sets to searches for SYS1 data sets. |
| TASKMAX | To specify an upper limit to the number of service tasks to process catalog requests. Once the limit is reached, new requests must wait. Setting an upper limit reduces the storage used by CAS. |
| UNALLOCATE | To close and unallocate a catalog from CAS, and to free up the CDSC or ISC storage used by the catalog. |
| VLF and NOVLF | To either assign a catalog to the catalog data space cache, or to prevent placement of additional records from a catalog into that cache. Records are not removed from the CDSC with these parameters. |
| WARNING and NOWARNING | Indicates if WTORS IDC1997I and IDC1998I will be issued for an IDCAMS DELETE UCAT FORCE or DELETE VVDS FORCE (WARNING), or if they will not be issued (NOWARNING). |

**Note:** This table does not reflect the detailed syntax of these commands. See "MODIFY CATALOG Command Syntax" on page 125 for syntax information.

## Starting and Stopping the Catalog Cache for a Catalog

When your system is initialized during IPL, catalogs are assigned as eligible for either the in-storage catalog cache or the catalog data space cache. These two caches have different performance benefits. Although a catalog might perform well in one type of cache under most circumstances, you might find occasion to remove a catalog from cache, or to move it to a different type of cache. If a catalog using the in-storage cache (ISC) receives a lot of update activity, you might want to remove it from cache until the update activity is finished.

You can move any catalog in the catalog data space to the in-storage catalog cache. Once removed from the catalog data space, a catalog automatically begins using the in-storage catalog cache, unless you use the NOISC parameter to prevent it. If you use the NOISC parameter, the catalog is not cached.

The objective of in-storage cache is to cache only those records which are read directly. Records will not be cached for other types of requests, and therefore in-storage cache is not then in effect. The ISC flag will be turned on the first time a direct read is done. This flag will remain on until that record is no longer in cache. At that point it will remain off until the next direct read to the catalog.

Since the COFVLF*xx* member of SYS1.PARMLIB controls which catalogs use the catalog data space cache, under normal conditions you cannot move a catalog from the in-storage catalog cache to the catalog data space. If a catalog is entered in the COFVLF*xx* member, and you have used MODIFY CATALOG,NOVLF to prevent it from using the catalog data space, then issuing the MODIFY CATALOG,VLF command returns the catalog to using the catalog data space.

You might need to move a user catalog from the in-storage catalog cache to the catalog data space cache. To do this, update the COFVLFxx member of SYS1.PAMRLIB, issue NOISC, and stop and restart the VLF with the updated version of the COFVLFxx member. Once these steps are completed without any errors, close the user catalog and issue MODIFY CATALOG, VLF. This allows CAS to use catalog data space cache for the user catalog.

Besides moving a catalog from one cache to the other, you can simply remove the catalog from cache. For example, if you determine that a catalog is not effectively using the catalog data space cache, you can remove it from the cache. The MODIFY CATALOG,REPORT,CACHE command provides information which you can use to evaluate cache performance.

Six parameters of the MODIFY CATALOG command can be used to modify how a catalog uses cache: ISC and NOISC; VLF and NOVLF; and ALLOCATE with NOISC or NOVLF. Which version you use depends on whether the catalog is already allocated to CAS.

If the catalog is not allocated to CAS, then you can allocate it to CAS with the MODIFY CATALOG,ALLOCATE command. The NOISC or NOVLF parameter can be used to prevent the catalog from using the specified cache. Otherwise, the catalog uses the last cache assigned to it (during IPL or a previous MODIFY CATALOG command).

If the catalog is already allocated and using ISC, the NOISC parameter can be used to terminate the ISC. Similarly, NOVLF can terminate the catalog data space.

If you want to assign a catalog in the CDSC to use ISC, close or unallocate it and use the NOVLF option. NOISC must *not* have been previously specified.

Removing catalogs from the ISC cache also reduces the amount of storage used by the catalog address space, probably at the expense of catalog performance.

## Changing the Multilevel Alias Search Level

A catalog alias can have up to four qualifiers. However, the actual number of qualifiers used, or the "multilevel alias search level", is initialized in the SYSCAT*xx* member of SYS1.NUCLEUS or LOAD*xx* member of SYS1.PARMLIB. You can

change this level without a system IPL by using the MODIFY CATALOG,ALIASLEVEL command. The change will remain in effect until the next time you IPL the system.

This command can be used to test various multilevel alias search levels. However, you should use this command with care. Changing the search level might result in some data sets becoming inaccessible, since the catalog searched might not be the catalog in which the data set is actually cataloged.

For example, the following aliases might be defined:

| Alias | For Catalog |
|---|---|
| APPLIC1.TESTING | SYS1.ICFCAT.TESTING |
| APPLIC1 | SYS1.ICFCAT.APPLIC |

In this case, if the original search level were 2, the data set APPLIC1.TESTING.DATA would be cataloged in SYS1.ICFCAT.TESTING. However, if you used MODIFY CATALOG,ALIASLEVEL(1) to change the search level to 1, subsequent requests for this data set would orient the catalog search to SYS1.ICFCAT.APPLIC. This results in a "data set not found" error.

See "The Multilevel Alias Facility" on page 19 for more information on the multilevel alias facility.

# Opening, Closing, Allocating, and Unallocating Catalogs

Using MODIFY CATALOG, you can specify that a catalog be closed, or that the catalog be allocated or unallocated from the catalog address space. You cannot, however, permanently close a catalog or prevent a user from accessing a catalog, nor can you explicitly open a catalog, with the MODIFY CATALOG command. To prevent users from accessing a catalog, use the access method services ALTER LOCK command.

Catalogs are allocated to the catalog address space when the first request for the catalog is processed. However, you can use the ALLOCATE parameter with the NOISC or NOVLF parameters to prevent the catalog from using the specified cache.

When you unallocate a catalog from CAS, the catalog is closed and all CAS private storage used by the catalog is freed. The device on which the catalog resides is also unallocated from CAS. However, CAS storage related to the catalog remains allocated. This space can only be freed by deleting the catalog. Common service area space for the catalog is not freed.

You can also close a catalog. When you use CLOSE, the catalog is closed but remains allocated to CAS. All the CAS private storage associated with the catalog is freed.

Closing or unallocating a catalog might be useful when the control blocks for a catalog become corrupted. When the catalog is opened again, new control blocks are built for the catalog.

If a catalog is using the CDSC, all CDSC space used by the catalog is freed when the catalog is closed or unallocated. This space is then available for the use of other catalogs in the CDSC.

If you alter a catalog's attributes (for example, the catalog's share options), you can close the catalog, and when it is reopened, the new values are recognized. Thus, catalog attributes can be changed without cancelling jobs or performing a system IPL.

If your installation has little or no catalog activity once your system is up and running, you might consider closing all the catalogs to free up CAS private storage associated with the closed catalogs.

You can also close a VVDS using the VCLOSE parameter. When you close a VVDS, the CAS private storage used by the VVDS is freed. The next request which uses the VVDS reopens the VVDS, and new control blocks are built for the VVDS.

Use VUNALLOCATE to unallocate all VVDSs. This might be necessary to allow the VARY command to vary a volume offline. VVDSs remain unallocated until you issue MODIFY CATALOG,NOVUNALLOCATE.

## Changing the Maximum Number of Catalogs and Tasks in CAS

When you IPL a system, the maximum number of catalogs that can be open in the catalog address space is set at 1024.   The maximum number of CAS service tasks is set to 180. The service task lower limit is specified in the SYSCAT*xx* member of SYS1.NUCLEUS or the LOAD*xx* member of SYS1.PARMLIB.

You can change either of these values using the CATMAX or TASKMAX parameters. Changing these values can help you limit the amount of storage used by CAS to perform catalog functions. The value you specify in either of these parameters is in hexadecimal.

When you use CATMAX to change the maximum number of catalogs that can be open in CAS and the new limit is lower than the previous limit, all open catalogs are closed. This does not unallocate catalogs. Catalogs remain allocated to CAS, but in restart status. All the storage associated with the catalogs that were closed is freed.

If a request for a closed catalog must be processed after the limit for open catalogs is reached, the least-recently used catalog is closed and the required catalog is opened.

Limiting the number of open catalogs affects catalog performance. However, if space is a primary consideration, you might need to set a maximum.

When you use TASKMAX to change the maximum number of CAS service tasks, first determine the current number of active CAS tasks. The REPORT parameter shows this value in the "CURRENT # SERVICE TASKS" field. The value you specify as the maximum should be at least as high as this value. If you specify a number lower than the current number of service tasks, the maximum number of service tasks allowed is set to the value of the current number of service tasks.

When the upper limit for tasks is reached, any new requests which require catalog resources wait until another task is finished. No user jobs fail because the limit is reached, but if the limit is set too low, it might cause a significant performance degradation for jobs on the system.

Catalog management can allocate up to 20 additional tasks for the exclusive use of catalog management. These additional tasks are only allocated if the maximum 180

tasks are in use, and catalog management requires the use of a CAS service task. Thus, the highest number of service tasks listed for the catalog address space might be as high as 200.

## Enabling and Disabling Operator Prompts for Certain Functions

If you delete a catalog using the IDCAMS DELETE UCAT FORCE command in an SMS environment, all data sets in the catalog are deleted. If you issue DELETE VVDS RECOVERY, all NVRS and VVRS from the VVDS are deleted. To prevent inadvertent deletion of data sets or VVR entries, IDCAMS issues a WTOR to the master console before proceeding with a delete command. Use MODIFY CATALOG,WARNING to enable prompting; it is the default. If you want to suppress the prompts, use MODIFY CATALOG,NOWARNING.

If the catalog address space is restarted by a MODIFY CATALOG,RESTART and the installation has suppressed prompting, it will attempt to preserve that setting after the restart. Restarting the catalog address space might reset the option to prompt. This occurs when certain catalog control blocks have been damaged and must be rebuilt as part of the restart.

The current setting of the option is displayed in response to MODIFY CATALOG,REPORT.

The installation may choose to force compliance with the documented syntax rules for data set names that are to be cataloged, or they may allow names that violate the rules to be created. The default is that checking will be enabled; data set names that violate the syntax rules will result in a failed catalog request. If you have programs that must create names that do not meet the syntax rules you can disable this checking; however IBM utilities may not be able to remove these entries from the catalog.

### Enabling and Disabling Data Set Name Validity Checking

The installation will either force compliance with the documented syntax rules for data set names that are to be cataloged, or it will allow names to be created that violate these rules. Data set names that violate the syntax rules will result in a failed catalog request. The default setting enables validity checking. If you have programs that must create names that do not meet the syntax rules, you can disable validity checking. However, IBM utilities may not be able to remove these entries from the catalog.

## MODIFY CATALOG Command Syntax

This section contains an explanation of the syntax and parameters of the MODIFY CATALOG command. Use this command to communicate with the catalog address space, in order to display information or request services.

When a system console operator issues any MODIFY CATALOG command except for MODIFY CATALOG,RESTART, messages return to that system console exclusively. If the system console operator issues the MODIFY CATALOG,RESTART, command, messages are returned to the master console and the console issuing the MODIFY CATALOG,RESTART. For information on system messages and abend codes, refer to *z/OS MVS System Messages, Vol 1 (ABA-AOM)* through *z/OS MVS System Messages, Vol 10 (IXC-IZP)*, and *z/OS MVS System Codes.*

| {MODIFY|F} | CATALOG |
|---|---|
| | **{ABEND(**task**)}** |
| | **{ALIASLEVEL(**n**)}** |
| | **{ALLOCATE(**catname**)[, {NOISC|NOVLF}]}** |
| | **{ALLOCATED[(**volser**)]}** |
| | **{CATMAX(**nnnn**)}** |
| | **{CLOSE(**catname**)}** |
| | **{DUMPON[(**rc**,**rsn**,**mm**[,**cnt**])]}** |
| | **{DUMPOFF}** |
| | **{ECSHR({ENABLE|REMOVE|STATUS},** catname**)}** |
| | **{ECSHR(AUTOADD)}** |
| | **{ECSHR({CONNECT|DISCONNECT|STATUS})}** |
| | **{ENABLE|DISABLE**(feature)**}** |
| | **{END(**id**)[,{REDRIVE|NOREDRIVE}]}** |
| | **{ENTRY[(**csectname**)]}** |
| | **{{ISC|NOISC}(**catname**)}** |
| | **{LIST[**(task)**]}** |
| | **{LISTJ[**(task),DETAIL**]}** |
| | **{REPORT}** |
| | **{REPORT,CACHE[(**catname**)]}** |
| | **{REPORT,DUMP}** |
| | **{REPORT,PERFORMANCE[(RESET)]}** |
| | **{RESTART}** |
| | **{{SYS%ON|SYS%OFF}}** |
| | **{TASKMAX(**nnn**)}** |
| | **{UNALLOCATE[(**catname**)]}** |
| | **{VCLOSE(**volser**)}** |
| | **{{VLF|NOVLF}(**catname**)}** |
| | **{VUNALLOCATE|NOVUNALLOCATE}** |
| | **{WARNING|NOWARNING}** |

where:

**ABEND(**task**)**
> specifies a CAS task to be abended with abend code 91A. Any catalog request in process at the time of the abend is redriven one time. The *task* identifier can be specified as:

> *id* To abend the task with this unique CAS identifier.

> *hextaskaddr*
>> To abend the task whose CAS service task has this four byte hexadecimal address.

> **ALLOCATE**
>> To terminate the CAS allocation task (IGG0CLGE) and attach a new CAS allocation task.

> **ANALYSIS**
>> To terminate the CAS analysis task (IGG0CLGG) and attach a new CAS analysis task.

> **MODIFY**
>> To terminate the CAS modify task (IGG0CLGA) and attach a new CAS modify task.

**ALIASLEVEL(**n**)**
> specifies the number of data set name qualifiers to be used in the multilevel

alias facility catalog search. The alias level is initially set at IPL with the value specified in the SYSCAT*xx* member of SYS1.NUCLEUS. The value *n* can be an integer between 1 and 4, inclusive.

**ALLOCATE(***catname***)[,{NOISC|NOVLF}]**
specifies that the catalog is to be allocated to CAS. The catalog must be specified in *catname*. The optional parameters are:

**NOISC**
specifies that the in-storage catalog is to be deactivated for the catalog.

**NOVLF**
specifies that the catalog data space cache is to be deactivated for the catalog.

**ALLOCATED[(***volser***)]**
specifies that the name, volume serial number, current allocation count, and status flags for every catalog currently allocated on the system is to be listed. This information is listed in message IEC348I.

If you specify a volume serial number (*volser*), only open catalogs which reside on the specified volume are listed.

**CATMAX(***nnnn***)**
specifies the maximum number of catalogs which can be opened concurrently in CAS. When the limit is reached, the least recently accessed catalog is closed, freeing the CAS storage it had occupied. Closed catalogs are not unallocated. They remain allocated, but in restart status with no CAS storage. If the new limit is less than the previous limit, all currently open catalogs are closed.

The maximum value is 9999.

The number specified for nnnn is in decimal.

**CLOSE(***catname***)**
specifies that the catalog named in *catname* is to be closed. All CAS storage for the catalog is released. The catalog is not permanently closed. The next job which requires the catalog opens it.

**DISABLE(***feature***)**
disables a particular optional feature. The only feature supported at this time is DSNCHECK, which indicates whether or not data set name syntax validation should be performed.

**{DUMPON|DUMPOFF}**
specifies whether or not CAS dynamic dumping is to occur. Dynamic dumping by CAS does not occur unless you specify DUMPON. You can use these DUMPON commands for problem solving:

```
MODIFY CATALOG,DUMPON
MODIFY CATALOG,DUMPON(rc,rsn,mm)
MODIFY CATALOG,DUMPON(rc,rsn,mm,cnt)
```

where:
**rc**   Specifies the catalog return code in decimal format (one to three characters to 255), or ***
**rsn**  Specifies the catalog reason code in decimal format (one to three characters with a maximum of 255), or ** asterisks for wild card searches. The catalog reason code in decimal (000-255), or ***
**mm**   Specifies the catalog module identifier in CAS, or **
**cnt**  Specifies the limit number in decimal (one to three characters to 999)

For example:

MODIFY CATALOG,DUMPON(8,42,**) creates a dump for any return code 8, reason code 42, and for any module that detected the error.

You can use the options in parentheses that follow the DUMPON parameter to create a dump whenever a given return code, reason code, and module identifier occur. The dump can prove valuable to service personnel for solving problems. Typically, the return code, reason code, and module identifier are available on return from CAS, and printed by IDCAMS. The module identifier corresponds to the last two characters in the catalog module name. For example, the module identifier is A3 for IGG0CLA3.

You can specify the return code, reason code, and module identifiers as a string of asterisks to indicate that any value encountered will match the value of that field. This is referred to as a generic match. All three fields cannot be specified simultaneously using asterisks. Whenever you specify a generic match for a field, it is assumed that field always matches the value that is returned by catalog for a catalog request. As an example:

```
MODIFY CATALOG,DUMPON(008,042,**)
```

creates a dump for return code 8, and reason code 42, regardless of the module that detects the error.

An option has been provided for a match count to obtain the *n*th occurrence of a return code, reason code, and module identifier. The match count is decremented by one each time a return code, reason code, and module identifier is set in the catalog address space. If this option is not specified or is set to 000, then the first occurrence causes a dump.

You can set only one set of return codes, reason codes, and module identifiers at a time. Each entry overwrites the previous information. After a match occurs, the information is cleared, and the original DUMPON status is maintained. If you enter DUMPON *without* the additional options, certain conditions produce dumps automatically. If you create a DUMPON with options, a match causes a dump and the return code, reason code, and module identifier are cleared. The DUMPON status remains on.

You can use MODIFY CATALOG,REPORT,DUMP to view the settings.

The header for the catalog dynamic dump contains the return code and reason code in hex. For example:

```
CAS DYNAMIC DUMP-IGG0CLA9 RCX'F6' RSNX'00'
```

**ECSHR({ENABLE|REMOVE|STATUS},*catname*}**

The named catalog should be enabled or removed from the ECS structure in the coupling facility or status for the catalog should be returned.

ENABLE causes the named catalog that is temporarily ineligible to be enabled to the ECS structure if all the following conditions are true:
- The catalog has the ECSHARING attribute
- The catalog has shareoptions(3 4)
- The catalog resides on a volume that is defined as shared
- The system is connected to the ECS structure

If any of the above conditions are not true, the request is rejected. The issuer of this command should ensure that all current or potential sharers of the catalog are capable of ECS. Otherwise, manual intervention might be required to return to the VVDS sharing protocol.

REMOVE causes the named catalog to be removed from the ECS structure, thereby affecting all systems sharing this catalog. The sharing protocol then reverts to the VVDS method. The MODIFY CATALOG command with the ECSHR(ENABLE,...) parameter must be used to re-enable this catalog for ECS activity. The catalog will not automatically be added (even if AUTOADD is enabled) without issuing this command.

STATUS causes message IEC380I to be displayed. It includes the status of the CF connection as well as the ECS status of the named catalog.

**ECSHR(AUTOADD)**

AUTOADD indicates whether ECS-eligible catalogs should be automatically added to the ECS structure.

AUTOADD causes ECS-eligible catalogs to be automatically added to the ECS structure on the next reference to the catalog. See the ENABLE parameter description above for the conditions that make a catalog eligible. AUTOADD is a sysplex-wide function.

ENABLEALL causes all temporarily ineligible catalogs known to CAS to be enabled for ECS.

**ECSHR({CONNECT|DISCONNECT|STATUS})**

indicates whether the system should connect to/disconnect from the ECS structure in the coupling facility or if the ECS system status should be returned.

CONNECT causes the system to connect to the ECS structure. The catalog address space issues an IXLCONN request to allocate and connect to the ECS structure or to connect to the already allocated structure.

DISCONNECT causes the system to disconnect from the ECS structure. The catalog address space issues an IXLDISC request. Any ECS catalogs in the structure are removed, and will subsequently be shared via the VVDS sharing mode.

STATUS causes message IEC380I to display. It includes the status of the CF connection as well as the status of each catalog that has been referenced since the last IPL.ENABLE{DISABLE|(DNSCHECK).

Data set names to be added to the catalog are syntax checked and the catalog request will fail if the data set name is not valid. This support will automatically be enabled at IPL. The syntax checking may be disabled by the MODIFY CATALOG,DISABLE(DSNCHECK) operator command. It may be enabled by the MODIFYCATALOG,ENABLE(DSNCHECK) operator command. The current setting of the option will persist across Catalog restarts. For more information on how to define the data set name, refer to *z/OS MVS JCL Reference*, and *z/OS DFSMS: Using Data Sets*.

The `MODIFY CATALOG,`ENABLE or DISABLE(DSNCHECK) command might have to be disabled if installation uses locally-written or OEM programs that create data set names that do not match syntax requirements. System programmers should review syntax guidelines.

Stricter data set name validation is enforced. If enabled, data set names are now syntax-checked for validity when a data set is added to a catalog. A user

can disable the validity checking using a new parameter on the `MODIFY CATALOG` command. Using validity checking will ensure that invalid names are not accepted.

This check will be performed on any data set name at the time it is to be added to the catalog (data set renames are processed as an add of the new name followed by a delete of the old, therefore, renames to an invalid data set name would also be prevented). The installation will have the option to disable this validity checking by a form of the MODIFY CATALOG operator command, but the default will be to perform the validity checks.

**ENABLE(***feature***)**
enables a particular optional feature. The only feature supported at this time is DSNCHECK, which indicates whether or not data set name syntax validation should be performed.

**END(***id***)[,{REDRIVE |NOREDRIVE}]**
specifies that the task identified by *id* is to be ended. This is the preferred method of ending a CAS task. The subparameters are:

*id*      specifies the unique task identification.

**REDRIVE**
specifies that any catalog request in process at the time of the abend is redriven an unlimited number of times. The CAS task is abended with abend code 91A.

**NOREDRIVE**
specifies that any catalog request in process at the time of the abend is failed with catalog return code 246. The CAS task is abended with abend code 71A.

**ENTRY[(***csectname***)]**
specifies that the starting addresses, the FMIDs, and the PTF/APAR levels of all the modules in the catalog load modules IGG0CLX0 (resident in CAS) and IGG0CLHA (resident in the link pack area) are to be displayed. The information is displayed in message IEC349I.

You can specify an individual CSECT in *csectname*, or you can have information for all entry points displayed by omitting a CSECT name.

The output of this command is probably best viewed on the system log due to its size, if all entry points are requested.

**{ISC|NOISC}(***catname***)**
specifies whether the indicated catalog's records are to be held in the in-storage catalog (ISC). The catalog must already be allocated to use this command. **ISC** specifies the catalog is to use ISC; **NOISC** specifies the catalog is not to use ISC.

The objective of in-storage cache is to cache only those records which are read directly. Records will not be cached for other types of requests, and therefore in-storage cache is not then in effect. The ISC flag will be turned on the first time a direct read is done. This flag will remain on until that record is no longer in cache. At that point, it will remain off until the next direct read to the catalog.

**LIST[(***task***)]**
specifies that currently active CAS service tasks, their related job names, elapsed times, and unique identifications, are to be listed. The information is listed in message IEC347I. The identifiers listed can then be used in other MODIFY CATALOG commands that require a CAS task ID.

All current tasks are listed unless *task* is specified. The *task* can be specified as:

*id*  To list information about the task with this unique identification.

*hextaskaddr*
   To list information about the task whose task control block (TCB) address is this four byte hexadecimal address.

*jobname*
   To list information about all service tasks currently active for this job.

**LISTJ(*jobname*)[,DETAIL]**
   Specifies that information about an active CAS service task should display. You can use the DETAIL keyword to request optional information. This optional information is primarily internal information about the catalog request. However, some of the information can be useful for diagnosing real-time problems involving the catalog address space.

   For example, an entry could show that the request has been waiting for a long time for completion of an ENQ. Then the ENQ resource name shown in the detail information could be used with a D GRS,C to find out what task in the system might be causing the wait condition.

**REPORT**
   Provides basic information about some of the current limits and installation-specified defaults that are selected for the catalog address space.

**REPORT,CACHE[(*catname*)]**
   Causes general information on catalog cache status to be listed. The report generated shows information useful in evaluating the catalog cache performance for the specified catalogs. If you do not specify a catalog name (*catname*), performance information for all catalogs currently active in the catalog address space is listed.

**REPORT,DUMP**
   The DUMP option is used to display the current dump status of catalog address space. This dump status can be enabled or disabled by DUMPON and DUMPOFF forms of the MODIFY command.

**REPORT,PERFORMANCE**
   PERFORMANCE lists information about events in CAS that invoke code outside of the catalog component. It shows the total number of occurrences of each event and the average time spent completing that event.

**REPORT,PERFORMANCE(RESET)**
   Use the RESET keyword to reset all performance statistics being accumulated. This allows you to periodically reset the statistical information and gather data to create a profile of the performance of the Catalog Address Space in your environment. The RESET keyword is also helpful in problem determination by allowing you to set the current statistics to zero during a period when performance might be a problem, and gather data specifically for that period.

**RESTART**
   specifies that CAS is to be restarted in a new address space. The CAS mother task is ended with abend code 81A, and any catalog requests in process at the time are redriven. The RESTART command should *only* be used when the only other option is an IPL. You should try other versions of the MODIFY command first before you use RESTART to solve catalog or VVDS problems.

**{SYS%ON|SYS%OFF}**

specifies that SYS% conversion to SYS1 either is (SYS%ON) or is not (SYS%OFF) to occur. The default setting for SYS% conversion is set during system initialization.

**TASKMAX(***nnn***)**

specifies the maximum number of CAS service tasks that are attached to process catalog requests. Once this limit has been reached, further requests for CAS services are delayed until a task control block becomes available.

The value *nnn* must be in the range of 24 to 180. The value must also be at least as high as the current number of tasks active in CAS. This value is displayed using the REPORT parameter.

If the value *nnn* is less than the current number of service tasks, the value for the maximum number of service tasks is set to the larger of 180 or the current number of service tasks.

The number specified for nnn is in decimal.

**UNALLOCATE[(***catname***)]**

specifies that the indicated catalog is to be removed from CAS. The CAS storage used by the catalog is freed, and if this is the only catalog that resides on the device, the device on which the catalog resides is unallocated from CAS.

If you do not specify a catalog, all catalogs allocated to CAS are unallocated.

**VCLOSE(***volser***)**

specifies that the VVDS which resides on the volume with the volume serial number *volser* is to be closed. The VVDS is reopened by the next request for the VVDS, and new control blocks are built.

Do not use VCLOSE to unallocate a VVDS to vary a volume offline. Use VUNALLOCATE.

**{VLF|NOVLF}(***catname***)**

specifies that the catalog data space cache is to be activated (VLF) or deactivated (NOVLF) for the catalog named *catname*. To activate the catalog data space cache (CDSC) for a catalog, the catalog must already be defined as eligible for CDSC. To deactivate CDSC, the catalog must already be allocated to CDSC.

**VUNALLOCATE|NOVUNALLOCATE**

specifies that all dynamically allocated VVDSs are to be unallocated from CAS (VUNALLOCATE) when a request is completed, or left allocated to CAS (NOVUNALLOCATE) after a request has completed. By default, VVDSs are unallocated after processing a request. Performance can be improved if you specify NOVUNALLOCATE, because repeated requests for a volume do not result in repeated dynamic allocations for the VVDS.

If you specify NOVUNALLOCATE, VVDSs remain allocated until VUNALLOCATE is issued, CAS is restarted, or the system is IPLed.

VUNALLOCATE can be used to unallocate a VVDS from CAS to allow a volume to be varied offline.

**WARNING|NOWARNING**

tells the system to issue message IDC1997I or IDC1998I before initiating an IDCAMS DELETE UCAT FORCE, or DELETE VVDS RECOVERY. The messages require an operator response to complete the commands.

The MODIFY CATALOG command displays, under the heading of DELETE UCAT/VVDS WARNING, the current setting of this option. For information about this type of IDCAMS DELETE, see *z/OS DFSMS Access Method Services*.

If you do not want the system to issue the messages, use NOWARNING.

# Appendix A. Catalog Diagnostic Information

This appendix contains diagnosis, modification, or tuning information.

## The Basic Catalog Structure (BCS)

The BCS is a VSAM key-sequenced data set and contains volume, data set security, ownership, and association information for VSAM and non-VSAM data sets.

A BCS can also point to a volume on which only non-VSAM data sets and generation data sets reside. If the volume is SMS-managed, the VVDS also contains information about the non-VSAM and generation data sets. If the volume is not SMS-managed, the information is only maintained in the BCS.

Related information in the BCS is grouped into logical, variable-length, spanned records related by key. The BCS uses keys that are the data set names (plus 1 character for extensions). A control interval can contain multiple BCS records. To reduce the number of I/Os necessary for catalog processing, logically related data is consolidated in the BCS.

The cell is the smallest block of information in the BCS and might contain the name, volume, owner, and association information for a catalog entry.

Entries for SMS-managed data sets contain an SMS subcell. This subcell contains the names for the storage, data, and management classes for the data set.

VSAM components must have the same SMS attributes as their associated base cluster. These components must also be cataloged in the same catalog as the base cluster, and this catalog must be the one SMS defines as the default catalog. (The default catalog is the catalog the system chooses using only the catalog aliases and the data set names, that is, the catalog chosen when you let the system choose the catalog, instead of directing the search yourself.)

Temporary SMS-managed VSAM data sets have VVRs in the VVDS, but they do not have BCS entries.

## BCS Records

There are two types of BCS records: the sphere record and the nonsphere record. A sphere record contains one or more components. The key length of any record is 45 bytes, consisting of a 44-byte data set or object name and a 1-byte pad character to indicate an extension record.

Records and components are a logically related group of cells. These logical groupings are physically adjacent in a sphere record. An example of a component is the index of a VSAM data set.

### Sphere Records
Sphere records might have related extension records. Extension records are created when the maximum record size of the BCS cannot contain a new component entry for a VSAM data set or alternate DEFINE USERCATALOG command. The default is 32400 bytes.

**135**

An alternate index or generation data group must be able to fit within an extension record.

An extension record is created when:

- An alternate index or generation data group is defined and does not fit in the current sphere record.
- A path is defined and the entry in the association cell does not fit in the sphere record.

  An extension cell will be created if there is a component entry for an alternate index or generation data set which can be moved into the extension record. The path entry must remain in the primary sphere record.
- Volumes are added to a cluster or alternate index and the volume cell does not fit in the sphere record.

The key of the extension record is the base cluster or generation data group name and the pad character.

The pad character is a binary number. The first extension is X'01', the second extension is X'02', and so on, up to the 240th extension which is X'F0'. The maximum number of extensions allowed is 240.

A component level entity is moved to the new extension record whether it is the component being updated or the last component on the current sphere record. For a VSAM sphere record, this is an alternate index. For a generation data group sphere record, the generation data set component is moved. Only one component resides in each extension record.

## Association Cells

Certain types of BCS entries can be paired with other BCS entries. This pairing of records is called an "association". Associated entries are connected by name and are indicated by an association cell in an entry.

Figure 7 illustrates an example of an association and its logical connections.
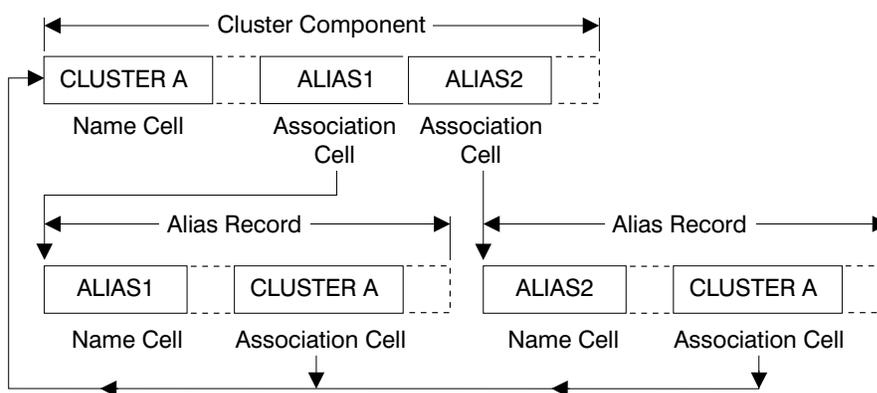


DA6C1004

*Figure 7. Example of an Association and Its Logical Connections*

The following associations can occur:

- Alias entries with catalog connector, non-VSAM data set, or generation data set entries.

- Catalog connector, non-VSAM data set, or generation data set entries with alias entries.
- Path entries with cluster or alternate index entries.
- Cluster or alternate index entries with path entries.
- Cluster truenames with data, index, and alternate index components.

## BCS Record Types

The first cell in each record has a cell type field, which is also the record type (or ID). The record type is identified by the DIAGNOSE command if DIAGNOSE lists the entry. The following are the possible record types and their one character identifiers:

| ID | Record Type |
|----|-------------|
| A  | non-VSAM data set |
| B  | generation data group |
| C  | cluster |
| D  | data component of a cluster |
| E  | VSAM extension record |
| G  | alternate index |
| H  | generation data set |
| I  | index component of a cluster |
| J  | generation data group extension cell |
| L  | library |
| R  | path |
| T  | true name |
| U  | user catalog connector |
| W  | volume |
| X  | alias |

# Initial Contents of a BCS

When the integrated catalog facility catalog is defined on a volume, a VSAM sphere record is built for the VVDS which resides on the volume with the BCS. This record has the name of the VVDS as its key (SYS1.VVDS.V*volser*).

The integrated catalog facility catalog has a sphere record similar to other VSAM key-sequenced data sets. This is a self-describing sphere record for the catalog, and contains information about the catalog itself. This sphere record is given a key of binary zeros to ensure it is the first record in the catalog. The data component name for the BCS is the catalog name as you have defined it. A true name record is created for the data and index components. The true name records are related with a key to the catalog name. A record is created to relate the index component to the catalog name of binary zeros, in the same way as for the data component.

# Allocation and Non-VSAM Catalog Entries

In non-VSAM data set catalog entries, there is a pointer to the data set's DSCB. This allows the system to locate the data set more quickly than can be done through a VTOC search. When a non-VSAM data set is allocated, if the pointer does not point to the correct DSCB, allocation marks the data set so that the data set can be recataloged when it is deallocated. When the data set is deallocated, it is recataloged with the information gathered at allocation time.

Normally, this maintains correct information in the catalog. However, if you move a data set or otherwise update the catalog entry for a data set, and allocation has

marked the data set to be recataloged at deallocation, the updates you made to the catalog entry are lost. Deallocation does not recognize that you have changed the entry.

For example, if you move a system data set, and allocation had marked that data set to be recataloged at deallocation time, the data set is recataloged with the volume serial number of the *old* volume, not the volume you moved it to.

There is no way to prevent this problem. To repair the catalog entry, simply delete the entry with DELETE NOSCRATCH, and recatalog the data set with DEFINE NONVSAM.

# The VSAM Volume Data Set (VVDS)

The VVDS is a VSAM entry-sequenced data set that has a 4KB control interval size. It contains information about the VSAM and SMS-managed non-VSAM data sets residing on the volume with the VVDS.

A VVDS is recognized by the restricted data set name SYS1.VVDS.V*volser*, where *volser* is the volume serial number of the volume on which the VVDS resides.

The VVDS is composed of a minimum of two records:
1. A VSAM volume control record (VVCR)
2. A VVDS self-describing volume record.

The first logical record in a VVDS is the VSAM volume control record (VVCR). It contains information for management of DASD space and the BCS names which currently have VSAM or SMS-managed non-VSAM data sets on the volume. It might have a pointer to an overflow VVCR.

The second logical record in the VVDS is the VVDS self-describing VVR (VSAM volume record). This self-describing VVR contains information that describes the VVDS.

The remaining logical records in the VVDS are VVRs for VSAM objects or non-VSAM volume records (NVRs) for SMS-managed non-VSAM data sets. The hexadecimal RBA of the record is used as its key or identifier.

Figure 8 shows the general structure of a VVDS.
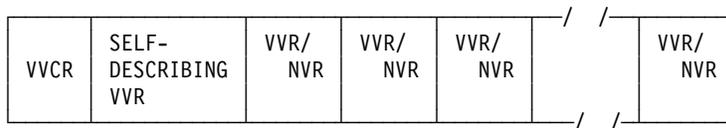


*Figure 8. VSAM Volume Data Set (VVDS) Structure*

# VSAM Volume Record (VVR)

VSAM volume records contain information about the VSAM data sets residing on the volume with the VVDS. If more than one VVR is associated with a component, the first (primary) VVR contains information pertaining to the data set as a whole. The other (secondary) VVRs do not repeat the primary information in their records but contain information for their own component, such as extents, RBAs, and allocation quantities, and most of the information needed to open a VSAM data set.

The number of VVRs for VSAM data sets varies according to the type of data set and the options specified for the data set. Table 16 contains the number of primary VVRs for each type of data set.

*Table 16. Number of Primary VVRs for Data Set Types*

| Data Set Type | Number of Primary VVRs |
|---|---|
| Entry-sequenced data set | 1 per volume for the data component |
| Key-sequenced data set with the NOIMBED option | 1 per volume for the data component; 1 for the index |
| Key-sequenced data set with the IMBED option | 1 per volume for the data component; 1 for the index; 1 per volume for the sequence set |
| Key range key-sequenced data set with the NOIMBED option | 1 per key range per volume; 1 for the index |
| Key range key-sequenced data set with the IMBED option | 2 per key range per volume; 1 per volume for the index |

The size of a VVR depends on its type (primary or secondary), and is determined by the combined lengths of VVR cells. Figure 9 shows the primary and secondary VVRs and the cells that constitute each VVR. These VVR cells are:

- VVR header cell
- VVR data set information cell
- VVR AMDSB cell
- VVR volume information cell

The VVRLEN field of the VVR header cell contains the length of the entire VVR. The VVRTYPE field of the same cell contains the VVR type code, which is either "Z" (for primary) or "Q" (for secondary).

`VVR Cells for Primary Volume VVR`

| VVR LEN | VVR HEADER CELL | VVR DATA SET INFO CELL | . SMS . SUBCELL . | VVR AMDSB CELL | VVR VOLUME CELL |
|---|---|---|---|---|---|

`VVR Cells for Secondary Volume VVR`

| VVR LEN | VVR HEADER CELL | VVR VOLUME CELL |
|---|---|---|

*Figure 9. VSAM Volume Record (VVR) Structure*

Figure 10 on page 140 shows examples of the information contained in each type of VVR cell.

**VVR Header Cell**

| LEN | TYPE | FLAG | KRQ | COMP NAME LEN | COMP NAME | CLS NAME LEN | CLS NAME | CAT NAME LEN | BASE CLS NAME LEN | BASE CLS NAME |
|---|---|---|---|---|---|---|---|---|---|---|

**VVR Data Set Information Cell**

| LEN | TYPE | ATTR1 | ATTR2 | OPEN IND | BUF SZ | PRI SPC | SEC SPC | SPC OPT | DS HU RBA | DS HA RBA | LRECL | RESV | EX CEI PT | DS HK RBA | CLS ATT | TIME STMP | ALTAB TIME STMP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**VVR Volume Information Cell**

| LEN | TYPE | VOL FG | #EXT | HKRBA | HURBA | HARBA | BLKSZ | BLK PER TRK | TRK PER AU | TYP EXT | TRK PER CYL | BYT PER TRK | BYT PER AU | LOW KEY LEN | LOW KEY | HI KEY LEN | HI KEY | EXT LEN | EXT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

*Figure 10. Examples of VVR Cell Information*

# Non-VSAM Volume Record (NVR)

The non-VSAM volume record (NVR) is equivalent to a VVR record, but the NVR record is for SMS-managed non-VSAM data sets. The NVR contains SMS-related information.

If an SMS-managed non-VSAM data set spans volumes, only the first volume contains an NVR for that data set. See Figure 11.

| NVR LEN | NVR HEADER CELL | NVR DATA SET INFORMATION CELL | . | SMS SUBCELL |
|---|---|---|---|---|

*Figure 11. Non-VSAM Volume Record (NVR) Structure*

# Appendix B. Catalog Search Interface User's Guide

Catalog Search Interface (CSI) is a read-only general-use programming interface that is used to obtain information about entries contained in ICF Catalogs. The ICF catalog entries are selected using a generic filter key provided as input. The generic filter key can be a fully-qualified entry name, in which case one entry is returned, or the generic filter key can contain "wild cards" so that multiple entries can be selected on a single invocation. The type or types of entries desired can also be specified. For instance, all non-VSAM entries that begin with "ABC" could be selected.

Field information for each entry is requested by specifying field names. This eliminates the need for the caller to know whether the information is in the Basic Catalog Structure (BCS) or in the VSAM Volume Data Set (VVDS).

A work area, whose address and size is provided on input, is used to return the selected entries and the field information for those entries. If all of the entries cannot be contained in the work area provided, then as many as possible are returned in one invocation, and an indicator is set to reflect that more entries exist. Subsequent invocations can specify that the request is being resumed and the additional entries can be obtained. Resumes can be repeated until all entries have been returned. The resume process allows the delivery of large amounts of information to the user program without impacting ICF Catalog resources.

## CSI Invocation

CSI can be invoked in either 24-bit or 31-bit addressing mode. CSI is reentrant and reusable. CSI can be invoked in any protection key and in either supervisor or problem state.The module name of the CSI program is IGGCSI00; it is stored in SYS1.LINKLIB.

The invocation can be done by any of the following methods:
- An assembler language CALL statement that results in a V-type address constant
- An assembler language LINK macro
- Assembler language LOAD and CALL macros
- A high-level language that results in one of these types of invocation.

When CSI is invoked:
- general-purpose register (GPR) 1 points to a parameter list
- GPR 13 points to a standard register save area of length 72 bytes
- GPR 14 contains the return address in the caller's program
- GPR 15 contains the entry point of CSI.

The parameter list pointed to by GPR 1 has three fullword entries:
- The first word of the parameter list contains a pointer to a 4-byte reason area used to return error or status information.
- The second word points to a list of selection criteria fields used to communicate with CSI.
- The third word points to a work area which contains data on return from CSI.

On return, a return code is passed in GPR 15 to indicate whether an error has occurred and the nature of the error. The first address in the parameter list points to a 4-byte reason area that contains information concerning the return code.

# The Parameter List

On invocation, general-purpose register 1 points to the CSI parameter list.

The first word in the parameter contains an address that points to a 4-byte reason area. On return from CSI, the reason area contains a module identification, reason code and return code. See the section on Output for further discussion of this parameter.

The second word contains the address of the selection criteria fields. The selection criteria data fields supply information to CSI on invocation, and contain information for a resume if there is more information to be returned than can fit in the user-provided work area.

The third word contains the address of a user work area in which entry information will be returned. The caller must place the size of the work area in the first word of the work area. Although this area has no fixed size, it must be between 1024 bytes and 1,048,575 bytes, inclusive. CSI will return the minimum size necessary to contain one entry, its catalog and fields included, and a reason/return code will be set if a larger work area is needed.

```
GPR1
┌─────────┐
│ Address │
└─────────┘
     │
     ↓
┌───────────┬───────────┬───────────┐
│ Address 1 │ Address 2 │ Address 3 │
└───────────┴───────────┴───────────┘
     │           │           │
     ↓           ↓           ↓
  ┌────────┐  ┌──────────┐  ┌────────┐
  │ Reason │  │ CSIFILTK │  │  Work  │
  │  Area  │  │ CSICATNM │  │  Area  │
  └────────┘  │ CSIRESNM │  └────────┘
              │ CSIDTYPS │
              │ CSIOPTS  │
              │ CSINUMEN │
              │ CSIENTS  │
              │    .     │
              │    .     │
              │    .     │
              └──────────┘
```
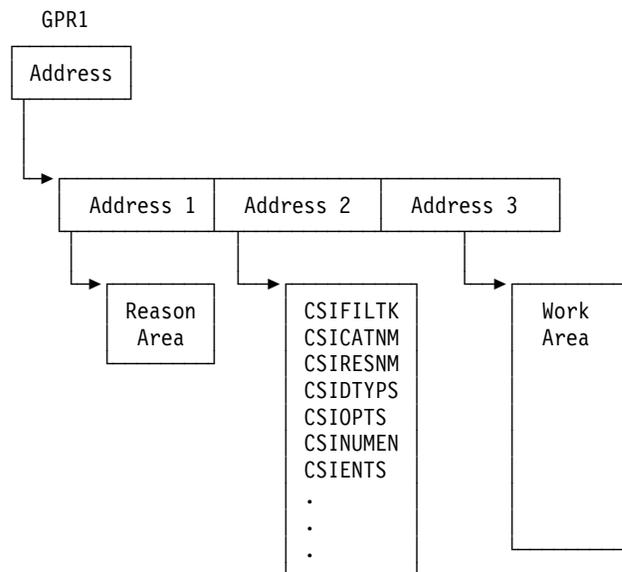
# Selection Criteria Fields

Table 17 shows the selection criteria fields.

*Table 17. Selection Criteria Fields*

| Offset | Type | Length | Field Name | Description |
|--------|------|--------|------------|-------------|
| 0(0) | | VL | CSIFIELD | CSI selection criteria fields |
| 0(0) | Character | 44 | CSIFILTK | Generic filter key |
| 44(2C) | Character | 44 | CSICATNM | Catalog name or blanks |
| 88(58) | Character | 44 | CSIRESNM | Resume name or blanks |
| 132(84) | Character | 16 | CSIDTYPS | Entry types |

*Table 17. Selection Criteria Fields  (continued)*

| Offset | Type | Length | Field Name | Description |
|---|---|---|---|---|
| 132(84) | Character | 1 | CSISTYPS | Entry types to be returned. All types = blanks |
| 148(94) | Character | 4 | CSIOPTS | CSI Options |
| 148(94) | Character | 1 | CSICLDI | Return data or index, Y or blank |
| 149(95) | Character | 1 | CSIRESUM | Resume, Y or blank |
| 150(96) | Character | 1 | CSIS1CAT | Search 1 catalog only, Y or blank |
| 151(97) | Character | 1 | CSIOPTNS | An F entry means to use fullword lengths; any other entry means use halfword lengths. |
| 152(98) | Fixed | 2 | CSINUMEN | Number of entries in table |
| 154(9A) | Character | VL | CSIENTS | Variable length table containing field names |
| 154(9A) | Character | 8 | CSIFLDNM | Field name (1st one) |

A PL/I mapping is provided in IGGCSINP.   An assembler macro is provided in
IGGCSINA. The fields are described in the following sections.

## CSIFILTK, Generic Filter Key

CSI uses a generic filter key supplied in CSIFILTK. A generic filter key is a
character string that describes the catalog entry names for which you want
information returned. The generic filter key can contain the following symbols and
are interpreted as follows:

* A single asterisk by itself indicates that either a qualifier or one or more
characters within a qualifier can occupy that position. An asterisk can
precede or follow a set of characters.

** A double asterisk indicates that zero or more qualifiers can occupy that
position. A double asterisk cannot precede or follow any characters; it must
be preceded or followed by either a period or a blank.

% A single percent sign by itself indicates that exactly one alphanumeric or
national character can occupy that position.

%%... One to eight percent signs can be specified in each qualifier.

If an absolute GDS name is specified followed by a period and one of the previous
symbols, the GDS name (if it exists) will not be returned from the search. However,
it is possible that longer data set names that start with the same prefix as the
absolute GDS name will be found. See the CSDFILTK examples that follow.

## CSIFILTK Examples

```
        VSAM.DATA.SET%
   will return entry names:
                        VSAM.DATA.SET1
                        VSAM.DATA.SET2
      will not return:
                        VSAM.DATA.SET30
        VSAM.DATA.SET%%
   will return entry names:
                        VSAM.DATA.SET30
                        VSAM.DATA.SET31
      will not return:
```

```
                                       VSAM.DATA.SET1
                                       VSAM.DATA.SET2
         VSAM.*.SET
         will return entry names:
                                       VSAM.DATA1.SET
                                       VSAM.DATA2.SET
             will not return:
                                       VSAM.DATA.SET.KSDS
         VSAM.*A
         will return entry names:
                                       VSAM.A
                                       VSAM.BA
                                       VSAM.BBA
             will not return:
                                       VSAM.B
                                       VSAM.AB
         VSAM.DATA.*
         will return entry names:
                                       VSAM.DATA.SET1
                                       VSAM.DATA.SET2
             will not return:
                                       VSAM.DATA.SET.KSDS
         VSAM.DATA*
         will return entry names:
                                       VSAM.DATA1
                                       VSAM.DATA23
             will not return:
                                       VSAM.DATA.SET
         VSAM.**
         will return entry names:
                                       VSAM
                                       VSAM.DATA.SET1
                                       VSAM.DATA.SET2
                                       VSAM.DATA.SET.KSDS
             will not return:
                                       VSAM1.DATA.SET
         VSAM.DATA.SET
         will return entry name:
                                       VSAM.DATA.SET   only
         **.DATA
         will return entry names whose low level qualifier is DATA:
                                       VSAM.DATA
                                       NONVSAM.WORK.DATA
         **
         will return every entry name in a catalog:
```

For a GDG base named DATASET.GDG containing the following GDS entries:

```
DATASET.GDG.G0001V00
DATASET.GDG.G0002V00
DATASET.GDG.G0003V00
```

and for the non-VSAM data set named DATASET.GDG.G0001V00.XYZ, the following keys will return the following results:

```
DATASET.GDG.**
     DATASET.GDG.G0001V00
     DATASET.GDG.G0002V00
     DATASET.GDG.G0003V00
     DATASET.GDG.G0001V00.XYZ

 DATASET.GDG.G0001V00
     DATASET.GDG.G0001V00

 DATASET.GDG.G0001V00.**
     DATASET.GDG.G0001V00.XYZ
```

```
DATASET.GDG.G000%V00
    DATASET.GDG.G0001V00
    DATASET.GDG.G0002V00
    DATASET.GDG.G0003V00

DATASET.GDG.G000%V00.**
    DATASET.GDG.G0001V00
    DATASET.GDG.G0002V00
    DATASET.GDG.G0003V00
    DATASET.GDG.G0001V00.XYZ
```

**Exception:** The entries returned by the data set name are not necessarily returned in ascending order.

## CSICATNM, Catalog Name

CSICATNM is used for catalog selection. The following paragraphs describe how CSI performs catalog selection.

CSI will use the catalog name supplied in CSICATNM to search for entries if CSICATNM is not blanks. If CSICATNM is blanks, Catalog Management will attempt to use the high-level qualifier of CSIFILTK to locate an alias that matches. If an alias is found, the user catalog for that alias will be searched. Otherwise, the master catalog will be searched.

Caution should be exercised in using high-level qualifiers that contain generic filters because multiple catalogs can be searched. A high-level of ″**″ will cause a search of every user catalog in the system.

CSIS1CAT can be used to limit the search to one catalog only. This is useful if the filter key could cause catalog management to select more than one catalog for searching. Also if not set, the master catalog will be searched in addition to one selected user catalog. In cases where the catalog name is supplied as input and a resume is done, CSIS1CAT will cause only that catalog whose name is supplied as input to be searched. Otherwise, on resume, catalog management cannot tell whether this is a search across many catalogs and the resume caused the CSICATNM to be set or if the name was supplied by the caller.

If a tape volume catalog library entry type or a tape volume catalog volume entry type is specified in CSIDTYPS, a tape volume catalog will be searched. Tape volume catalog library entry types and tape volume catalog volume entry types should not be mixed with ICF Catalog entry types.

## CSIRESNM, Resume Name

CSIRESNM is the name of the catalog entry on which the search will resume if the work area space for return information is used up. If the request can be resumed, CSIRESUM will be set to ″Y″ on return from the call to CSI, CSIRESNM will contain the name of the entry on which to resume, and CSICATNM will contain the name of the catalog in which the entry to be resumed was found. Normally, the application program will test CSIRESUM, and if it is ″Y″, the application program will reissue the call to CSI without changing CSIRESNM and CSICATNM.

## CSIDTYPS, Entry Types

CSIDTYPS determines what type of catalog entries will be returned.

Valid types for CSIDTYPS are:

**Type    Description**

| | |
|---|---|
| **A** | non-VSAM data set |
| **B** | Generation data group |
| **C** | Cluster |
| **G** | Alternate index |
| **H** | Generation data set |
| **R** | VSAM path |
| **X** | Alias |
| **U** | User catalog connector entry |
| **L** | Tape volume catalog library entry |
| **W** | Tape volume catalog volume entry |

VSAM components, data and index, are returned with the cluster. Thus, there are no type specifications for them, however, ″D″ and ″I″ types will appear in the output information.

The valid types can be mixed and in any order. Blanks cannot separate the types. For instance, "ABH" might be specified to get only the non-VSAM, generation data group and generation data set entries. All other positions in CSIDTYPS must be blanks (X'40') when the types are specified.

All blanks for CSIDTYPS can be set and will get types A, B, C, G, H, R, X, U. These are the ICF Catalog types. L and W must be explicitly specified in order to get the Tape Volume Catalog entries.

# CSIOPTS, Options

### CSICLDI
CSICLDI determines whether information will be returned if the data and index names of a cluster do not match the filter key, but the cluster name does. If CSICLDI is set to ″Y″ then components are returned if the cluster name matches the filter key.

### CSIOPTNS
If this value is set to F, all fields used to describe the output data for each entry will be fullword (4 bytes) in length. If the value is set to blank, all fields will be halfword (2 bytes) in length. Note that using 2–byte fields might mean that certain types of entries that return more than 65,535 characters for that catalog entry will not be processed.

### CSIRESUM
CSIRESUM is set to ″Y″ if CSI detects that there are more catalog entries that meet the search criteria. This field must be blank on the first invocation. CSIRESUM can be tested, and CSI can be called again to obtain more entries. CSIRESUM will be set to blank (X'40) if no more entries meet the search criteria.

### CSIS1CAT
CSIS1CAT causes only one catalog to be searched. It is used in conjunction with CSICATNM to determine which catalogs to search. Refer to the section on CSICATNM for more details concerning catalog selection.

# CSINUMEN, Number of Field Names

CSINUMEN is a binary number indicating the number of field names following in the subscripted area CSIFLDNM.

There is a limit of 100 field names per invocation of CSI. CSINUMEN cannot be greater than 100.

## CSIFLDNM, Field Names

CSIFLDNM is a list of 8-byte field names. If the field name is not eight characters long, then it must be padded on the right with blanks to make eight characters.

Valid field names that can be used in the list and the information returned for each field name is described in section "Field Name Directory" on page 154.

## Return Codes for General Purpose Register 15

On return, general-purpose register 15 can contain the following:

| Return Code | Description |
|---|---|
| **0** | No errors or return messages |
| **4** | Information was returned from Catalog Management processing. Further information is returned in the 4-byte reason area pointed to by the first address in the parameter list. See Table 18 on page 148 for the interpretation of this information. |
| **8** | Failure in Catalog Search Interface routine. Further information is returned in the 4-byte reason area pointed to by the first address in the parameter list. See "Return Code 8" on page 148 for the interpretation of this information. |
| **C** | Error in Catalog Search Interface routine parameter list - check for zero entries. |
| **10** | Parameter list pointer in general-purpose register 1 is zero. |

## Return Codes 4 and 8

When general-purpose register 15 contains 4 or 8, the first address in the parameter list points to a 4-byte reason area. The pattern of the information in the reason area is:

| Description | Length in Bytes |
|---|---|
| **Module ID** | 2 |
| **Reason code** | 1 |
| **Return code** | 1 |

### Return Code 4

When general-purpose register 15 contains 4, the reason area information is passed back from Catalog Management.

The module identification is a two-character EBCDIC code that can be used by IBM Service Personnel to determine which Catalog Management module set the reason and return code. The module identification information is not generally useful for applications.

Reason and return codes returned by Catalog Management are found in LookAt under message IDC3009I. For a description of LookAt, see *z/OS MVS System Messages, Vol 3 (ASB-BPX)*. Refer to this book for the description of a particular return and reason code Return codes 100 (X'64') and 122 (X'7A') are not documented in this book. Table 18 on page 148 shows the meaning of these return codes.

*Table 18. Return Codes 100 and 122*

| Return Code | Description | Programmer Response |
|---|---|---|
| 100 | Catalog Management has detected an error while processing the request.<br>• Reason Code 4: at least one data set entry is returned with an error.<br>• Reason Code 8: at least one catalog entry is returned with an error. | • Reason Code 4: locate the entry or entries with the CSIENTER flag set and inspect the CSIRETN field for further information.<br>• Reason Code 8: inspect the return code and reason code for the catalog entry for further information. |
| 122 | An invalid filter key was provided. | For all reason codes: fix filter key and resubmit.<br>**Note:** This return code may also indicate an invalid data set name in the catalog being searched. |

### Return Code 8
When general-purpose register 15 contains 8, the reason area information is passed back from Catalog Search Interface routine. The module identification is always set to X'FFFF'. The reason codes have the following meanings:

**01**  Insufficient storage for Getmain, increase region size

**02**  Invalid entry type in CSIDTYPS

**03**  Invalid data/index option in CSICLDI, should be Y or blank

**04**  Invalid resume option in CSIRESUM, should be Y or blank

**05**  Invalid search one catalog option in CSIS1CAT, should be Y or blank

**06**  Invalid number of fields value in CSINUMEN, should be between zero and 100, inclusive

**07**  Invalid work area length, should be between 1024 and 1048575, inclusive

**08**  The CSIOPTNS value is not F or blank

The return code is set to 8 for all reason codes.

# Return Work Area Format

Entry information retrieved will be returned in the user-provided work area. Resume information is returned in the Selection Criteria Fields. If on return CSIRESUM is set to ″Y″, then CSIRESNM will be set to the next entry to be processed, and CSICATNM will be the catalog in which that entry will be found. When the program has processed returned entry information, it should re-invoke CSI to resume at the next entry.

The caller must set CSIUSRLN prior to CSI invocation. CSIUSRLN is the size in bytes of the work area including itself. CSIUSRLN must be a fixed fullword value between 1024 and 1,048,575, inclusive. Although CSIUSRLN can be 1,048,575, large sizes may impact Catalog Management resources. A size of 64,000 is generally recommended.

All other data is returned by CSI in the user-provided work area.

# Work Area Format Table

Upon return to the caller, the work area will be in the format shown in Table 19.

*Table 19. Work Area Format Table*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| **INFORMATION RETURNED FOR WORK AREA** | | | | |
| 0(0) | Character | VL | CSIRWORK | CSI Return Work Area |
| 0(0) | Fixed | 4 | CSIUSRLN | Total length of work area. User provided. |
| 4(4) | Fixed | 4 | CSIREQLN | Minimum required work area for 1 catalog name entry and 1 data entry entry. |
| 8(8) | Fixed | 4 | CSIUSDLN | Total length of work area used in returning entries. |
| 12(C) | Fixed | 2 | CSINUMFD | Number of field names plus 1. |
| **INFORMATION RETURNED FOR EACH CATALOG** | | | | |
| 0(0) | Bitstring | 1 | CSICFLG | Catalog flag information |
| | 1... .... | | CSINTICF | Non-ICF catalog, not supported. |
| | .1.. .... | | CSINOENT | No entry found for this catalog. |
| | ..1. .... | | CSINTCMP | Data gotten for this catalog is not complete. |
| | ...1 .... | | CSICERR | Whole catalog not processed due to error. |
| | .... 1... | | CSICERRP | Catalog partially processed due to error. |
| | .... .111 | | | Reserved. |
| 1(1) | Character | 1 | CSICTYPE | Catalog type. X'F0' |
| 2(2) | Character | 44 | CSICNAME | Catalog name |
| 46(2E) | Character | 0 | CSICRETN | Return information for Catalog. |
| 46(2E) | Character | 2 | CSICRETM | Catalog return module ID |
| 48(30) | Fixed | 1 | CSICRETR | Catalog return reason code |
| 49(31) | Fixed | 1 | CSICRETC | Catalog return code |
| **INFORMATION RETURNED FOR EACH ENTRY** | | | | |
| 0(0) | Bitstring | 1 | CSIEFLAG | Entry flag information. |
| | 1... .... | | CSIPMENT | Primary entry. |
| | 0... .... | | | This entry associates with the preceding primary entry. |
| | .1.. .... | | CSIENTER | Error indication is set for this entry and error code follows CSIENAME. |
| | ..1. .... | | CSIEDATA | Data is returned for this entry. |
| | ...1 1111 | | | Reserved. |
| 1(1) | Character | 1 | CSIETYPE | Entry Type.<br>A Non-VSAM data set<br>B Generation data group<br>C Cluster<br>D Data component<br>G Alternate index<br>H Generation data set<br>I Index component<br>R Path<br>X Alias<br>U User catalog connector entry<br>L ATL Library entry<br>W ATL Volume entry |

*Table 19. Work Area Format Table  (continued)*

| Offset | Type | Length | Name | Description |
|---|---|---|---|---|
| 2(2) | Character | 44 | CSIENAME | Entry name. |
| 46(2E) | Character | 0 | CSIERETN | Error return information for entry. Only exists if CSIENTER is 1. |
| 46(2E) | Character | 2 | CSIERETM | Entry return module ID |
| 48(30) | Fixed | 1 | CSIERETR | Entry return reason code |
| 49(31) | Fixed | 1 | CSIERETC | Entry return code |
| 46(2E) | Character | VL | CSIEDATA | Returned data for entry. Only exists if CSIENTER is 0. |
| **IF CSIOPTNS IS NOT F:** | | | | |
| 46(2E) | Character | 2 | CSITOTLN | Total length of returned information including this field and length fields. The next entry begins at this offset plus this length. |
| 48(30) | | 2 | | Reserved. |
| 50(32) | Character | VL | CSILENFD | Length of fields. There is one length field returned for each field name passed on input. |
| 50(32) | Character | 2 | CSILENF1 | First length field. |
| **IF CSIOPTNS IS F:** | | | | |
| 46(2E) | Character | 4 | CSITOTLN | Total length of returned information including this field and length fields. The next entry begins at this offset plus this length. |
| 48(30) | | 4 | | Reserved. |
| 50(32) | Character | VL | CSILENFD | Length of fields. There is one length field returned for each field name passed on input. |
| 50(32) | Character | 4 | CSILENF1 | First length field. |
| **INFORMATION RETURNED FOR EACH FIELD NAME** | | | | |
| 0(0) | Character | VL | CSIFDDAT | Field data. For each field name passed on input, there will be a data item corresponding to its length. |
| | | | | The next entry would begin here if more than 1 entry is returned. |
| | | | | The next catalog entry would begin after all of the entries if more than 1 catalog is searched. |

# Work Area Format Picture



```
                          ── W-O-R-K-A-R-E-A ──
4 bytes  ──▶  │ User provided work area length │  ◀── The caller must
                                                        set this length
4 bytes  ──▶  │ Required work area length      │       prior to the call

4 bytes  ──▶  │ Used user work area length     │

2 bytes  ──▶  │ Number of field names +1       │

              │ Flag │ Type │ Catalog name     │  ◀── 46 bytes - catalog
                                                        type is 'F0'
This field ─▶ │ Module ID │ RSN │ RC           │  ◀── 4 bytes
always
exists for    │ Flag │ Type │ Entry name       │  ◀── 46 bytes - 1 byte
a catalog                                               flag & type - 44
name entry    │ Module ID │ RSN │ RC           │       name

2 or 4   ──▶  │ Tot Len │ Reserved │ Length flds│  ◀── number of length
bytes                                                   fields equals to
of data                                                 number of field
length        │        Field Data             │        names supplied
including
itself and    │ Flag │ Type │ Entry name       │  ◀── Data set name,
length                                                  Alias, GDG, etc.
fields        │ Module ID │ RSN │ RC           │

              │ Tot Len │ Reserved │ Length flds│

              │        Field Data             │

              │ Flag │ Type │ Entry name       │  ◀── Data set name,
                                                        Alias, GDG, etc.
This field ─▶ │ Module ID │ RSN │ RC           │  ◀── 4 bytes
exists only
if CSIENTER   │ Tot Len │ Reserved │ Length flds│
is set
              │        Field Data             │  ◀── variable length
                                                        data
                                                        returned for
                                                        field names
              │ Flag │ Type │ Catalog name     │  ◀── Only appears if
                                                        filter search yields
              │ Module ID │ RSN │ RC           │       multiple catalogs

              │ Flag │ Type │ Entry name       │  ◀── Data set name,
                                                        Alias, GDG, etc.
              │ Module ID │ RSN │ RC           │

              │ Tot Len │ Reserved │ Lengthflds │

              │        Field Data             │
```

# Work Area Format Description

The first field is the length of the work area and is set by the user to tell CSI how much data can be returned.

The second field is the minimum required length for a catalog entry and one entry's worth of returned data. If the minimum length is greater than the work area length,

then the work area length must be increased to at least as much as the minimum length. If this is not done, and a resume condition occurs, the user program will appear as if in an endless loop because the same information will be returned for each resume until the first length is increased to contain the entire entry. A generation data group (GDG) with all of its associated generation data sets (GDS) is seen by CSI as one record. Thus the required length for this entry is apt to be large.

The third field is the amount of space that was used in the work area. CSI always returns a full entry. If the last entry will not fully fit in the remaining work area space, the resume flag is set and the space at the end of the work area is unused. The unused space will usually be small.

Next follows the number of field names plus 1.

A catalog name entry is returned for every catalog processed. A catalog entry may be identified because its type is x'F0'. This is an artificial type invented so that the next catalog entry can be found. The catalog entry is always followed by return information. The return code portion will be zeroes if no problems were encountered while processing the catalog during the call.

Following the catalog entry is one or more entries contained in the catalog that match the search criteria (filter key). Each entry has flags, followed by its type and name. If the flags indicate, a module id, reason code and return code follow the entry name; otherwise, the field information for the entry follows.The flag byte in front of the entry type will also indicate possible errors that were encountered.

The ″MODULE ID / RSN / RC″ returned in the work area is returned when an error is detected by Catalog Management. This field only exists when the flag CSIENTER is set for this entry; otherwise, it is not present. See the previous section entitled ″Return Codes″, subsection ″Return Code 4″, for a description of this information.

If no errors or messages occurred, then field information for the entry is returned as a set of lengths and then the data corresponding to the lengths.

The first length field is the length of all of the returned data for this entry. It is the total of the length of the field itself, all of the length fields following it (one per fieldname supplied in CSIFLDNM), and the length of the actual data returned for this entry. The total length field is two bytes long, unless CSIOPTNS was specified as F, in which case this length field (and all remaining length fields) are four bytes long. The reserved field following this total length is also two or four bytes long.

Next is a set of lengths corresponding to the number of fields passed in.The length fields are two bytes long if CSIOPTNS is blank and four bytes long if CSIOPTNS is F. Each length is used to determine the length and position of the returned data for the entry.

For example, if three field names were supplied on input, then there will be three field lengths. Each length will be for the data immediately following the lengths. If the lengths had values 4, 6, and 8, then following the last length, there would be 4-bytes worth of data for the first field, 4-bytes from the last length field would be 6 bytes of data for the second field, and 10 bytes from the last length would be 8 bytes worth of data for the third field. Each length is set to:

- -1: If the data to be retrieved is suppressed; security data will be suppressed if the caller does not have the proper RACF authority.

- 0: If no data was found for this entry. This can happen, for instance, if the data does not apply to the particular entry type being supplied.
- 2 or 4: If the data field to be retrieved has variable length and does not exist. Data with variable length is always returned with two- or four-byte length information preceding the data (depending on the setting of CSIOPTNS) and is included as part of data. Therefore, when the field to be retrieved does not exist, this preceding length information is set to 0 and the total length of the data returned will then be the length of this preceding length information.
- n: If the data field to be retrieved has fixed length and does not exist. n is the length of the fixed length data and the data is set to be all 'FF'X.
- n: If the data is retrieved and is the total length of the data retrieved.

The following illustration shows the general relationship of the length fields and their corresponding data.
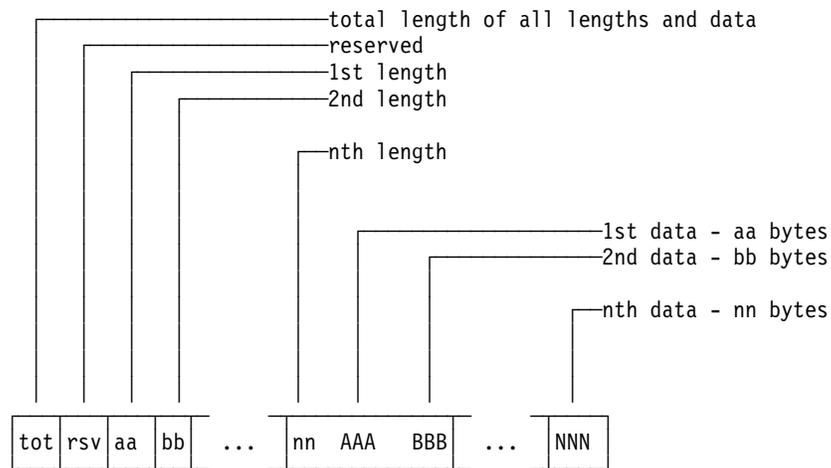


**Restriction:** Total information returned for an entry cannot exceed 65,535 characters unless CSIOPTNS is set to F (use fullword lengths). If CSIOPTNS is not set to F, any attempt to retrieve information greater than 65,535 characters results in setting the CSIENTER flag in that entry and the CSI request will continue with the next data set (if any). When the CSI request is complete, the return code will be 100 and the reason code will be 4, unless a more serious error occurs. For the entry with the CSIENTER flag set, the CSIERETN value is returned; the return code is 44 and the reason code is 14. For example, a request for the NAME fieldname of a user catalog (or data set) that has more than 1489 aliases defined would result in this error.

# Field Name Directory

These are valid field names that can be used in CSIFLDNM. The information returned for each field name is given in the description.

The REP column refers to fields that can repeat when returned by CSI.

# ICF Catalog Field Names

Table 20 shows the ICF catalog field names.

*Table 20. ICF Catalog Field Names*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| no | Character | 36 | ACTOKEN | Active compression dictionary token |
| no | Fixed | 8 | AMDCIREC | Control interval size for 4 bytes and maximum record size for 4 bytes |
| no | Fixed | 4 | AMDKEY | Relative position of KSDS key for 2 bytes and key length of KSDS key for 2 bytes |
| no | Bitstring | 1 | ATTR1 | Attributes: |
| | 1... .... | | | Speed |
| | .1.. .... | | | Unique |
| | ..1. .... | | | Reusable |
| | ...1 .... | | | Erase |
| | .... 1... | | | Not defined |
| | .... .1.. | | | Inhibit update |
| | .... ..1. | | | Temporary export |
| | .... ...1 | | | Track overflow |
| no | Bitstring | 1 | ATTR2 | Share attributes |
| | 11.. .... | | | Region (00 = 1, 01 = 2, 10 = 3, 11 = 4) |
| | ..11.... | | | System (00 = 1, 01 = 2, 10 = 3, 11 = 4) |
| | ....1111 | | | Not defined |
| no | Fixed | 4 | BUFSIZE | Maximum buffer size |
| no | Bitstring | 1 | CATTR | Attributes for pagespace and swapspace |
| | 1111 11.. | | | Not defined |
| | .... ..1. | | | Swap=1, noswap=0 |
| | .... ...1 | | | Data set is a pagespace |
| no | Bitstring | 1 | COMPIND | Compression indicator. |
| | 1..1 1111 | | | Not defined. |
| | .1.. .... | | | Data set is extended format |
| | ..1. .... | | | Data set is compressible |
| no | Fixed | 8 | COMUDSIZ | Compressed user data size |
| no | Character | VL | DATACLAS | SMS data class |
| yes | Fixed | 4 | DEVTYP | UCB device type |

*Table 20. ICF Catalog Field Names  (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| yes | Fixed | 3 | DSCBTTR | TTR of format-1 DSCB for non-VSAM data set |
| no | Mixed | 4 | DSCRDT2 | Creation date. Packed decimal YYDDDF for 3 bytes appended with one byte century indicator. If the century byte is 00 then add 1900 to get the year; if 01, add 2000. |
| no | Decimal | 3 | DSETCRDT | Creation date in format packed decimal YYDDDF |
| no | Decimal | 3 | DSETEXDT | Expiration date in format packed decimal YYDDDF |
| no | Mixed | 4 | DSEXDT2 | Expiration date. Packed decimal YYDDDF for 3 bytes appended with one byte century indicator. If the century byte is 00 then add 1900 to get the year; if 01, add 2000. |
| no | Character | 1 | ENTYPE | Entry type, ex., 'C' is cluster, 'A' is non-VSAM, etc. |
| no | Character | 44 | ENTNAME | The name of the entry |
| no | Character | 8 | EXCPEXIT | Exception exit |
| yes | Fixed | 2 | FILESEQ | File sequence number |
| no | Mixed | 4 | GDGALTDT | Last alteration date. Packed decimal YYDDDF for 3 bytes appended with one byte century indicator. If the century byte is 00 then add 1900 to get the year; if 01, add 2000. |
| no | Bitstring | 1 | GDGATTR | Generation data group attributes |
| | 0... .... | | | Delete oldest GDS when GATLIMIT exceeded |
| | 1... .... | | | Delete all GDSs when GATLIMIT exceeded |
| | .0.. .... | | | Do not scratch data set when rolled off |
| | .1.. .... | | | Scratch data set when rolled off if volume mounted |
| no | Fixed | 1 | GDGLIMIT | Maximum number of generation data sets allowed in the GDG |
| yes | Character | 4 | GENLEVEL | GDG generation level — 1 for each active generation in EBCDIC format c'0000' |
| yes | Fixed | 4 | HARBA | High-allocated RBA |
| no | Fixed | 4 | HARBADS | Data set high-allocated RBA |
| yes | Character | VL | HIKEYV | High Key on volume |
| yes | Fixed | 4 | HKRBA | RBA of data control interval with high key |
| yes | Fixed | 4 | HURBA | High-used RBA |
| no | Fixed | 4 | HURBADS | Data set high-used RBA |
| yes | Bitstring | 1 | ITYPEXT | Type of extent |

*Table 20. ICF Catalog Field Names  (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| | 1... .... | | | Sequence set with data, IMBED |
| | .1.. ... | | | Extents not preformatted |
| | ..1. .... | | | Converted VSAM data set volume |
| | ...1 1111 | | | Not defined |
| no | Binary | 1 | LOGPARMS | Value of LOG parameter set by IDCAMS DEFINE/ALTER<br>X'00' - never set<br>X'01' - LOG(NONE)<br>X'02' - LOG(UNDO)<br>X'03' - LOG(ALL) |
| no | Character | 26 | LOGSTRID | Value of LOGSTREAMID parameter set by IDCAMS DEFINE/ALTER |
| yes | Character | VL | LOKEYV | Low Key on volume |
| no | Fixed | 4 | LRECL | Average logical record size |
| no | Fixed | 8 | LTBACKDT | Last backup date |
| no | Character | VL | MGMTCLAS | SMS management class |
| yes | Character | 44 | NAME | The name of an associated entry |
| yes | Fixed | 2 | NOBLKTRK | Number of physical blocks per track (for pagespace data sets only) |
| yes | Fixed | 4 | NOBYTAU | Number of bytes per allocation unit |
| yes | Fixed | 4 | NOBYTTRK | Number of bytes per track |
| yes | Fixed | 1 | NOEXTNT | Number of extents |
| yes | Fixed | 2 | NOTRKAU | Number of tracks per allocation unit (for pagespace data sets only) |
| no | Character | 1 | NVSMATTR | Non-VSAM attribute information |
| | c'H' | | | Active GDS |
| | c'N' | | | Deferred GDS |
| | c'M' | | | Rolled-off GDS |
| | c'L' | | | Extended partitioned data set (PDSE) |
| | c'P' | | | POSIX data set |
| | x'00' | | | Simple non-VSAM data set |
| no | Bitstring | 1 | OPENIND | Open indicator |
| | .... ...1 | | | Open =1, note open=0, 1 may mean that the data set was not closed properly |
| no | Character | 8 | OWNERID | Owner of the data set |
| no | Fixed | 2 | PASSATMP | Number of attempts to prompt for password |
| no | Character | 8 | PASSPRMT | Password prompt code name |
| no | Character | 32 | PASSWORD | Four 8-byte passwords (VSAM data sets only) |
| yes | Fixed | 4 | PHYBLKSZ | Physical blocksize only |
| no | Fixed | 3 | PRIMSPAC | Primary space allocation |

*Table 20. ICF Catalog Field Names (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| no | Binary | 8 | RECVTIME | Recovery time, TOD value, local |
| no | Binary | 8 | RECVTIMG | Recovery time, TOD value, GMT |
| no | Bitstring | 1 | RGATTR | Alternate index attributes |
| | 1... .... | | | Upgrade=1, noupgrade=0 |
| | .1.. .... | | | Entry is an alternate index |
| | ..11 1111 | | | Not defined |
| no | Bit | 1 | RLSBWO | Value of BWO parameter set by IDCAMS DEFINE/ALTER<br>xxxx 0000 - undefined or not set<br>xxxx 0001 - BWO(TYPECICS)<br>xxxx 0010 - reserved<br>xxxx 0011 - BWO(TYPEIMS)<br>xxxx 0100 - reserved |
| no | Bit | 1 | RLSFLAGS | xxxx 0xxx - Recovery not required<br>xxxx 1xxx - Recovery required |
| no | Fixed | 3 | SCONSPAC | Secondary space allocation |
| no | Bitstring | 1 | SECFLAGS | Security flag information x'80' means the data set has a discrete RACF profile |
| no | xx.. .... | | SPACOPTN | Equals '01' for record allocation, '10' for track allocation, and '11' for cylinder |
| no | Character | VL | STORCLAS | SMS storage class |
| no | Fixed | 2 | STRIPCNT | Striping counts for striped data sets |
| yes | Character | 1 | TYPE | The type of an associated entry |
| no | Fixed | 8 | UDATASIZ | User data size |
| no | Character | 8 | USVRMDUL | User security verification module |
| no | Character | VL | USERAREC | User authorization record |
| yes | Character | 6 | VOLSER | Volume serial number |
| yes | xxx. .... | | VOLFLG | '100' is the primary volume with space allocated, '010' is the candidate volume with no space allocated, '001' is the overflow volume (keyrange data set only) with no space allocated |
| no | Bitstring | 1 | VSAMREUS | VSAM data set information |
| | 1... .... | | | Data set has RACF discrete profile |
| | .1.. .... | | | Index component data set |
| | ..1. .... | | | Reusable data set |
| | ...1 .... | | | Erase specified (cluster only) |
| | .... 11.. | | | Not defined |
| | .... ..1. | | | Swap space (cluster only) |
| | .... ...1 | | | Page space (cluster only) |
| no | fixed | 46 | VSAMSTAT | Statistics infomation for VSAM components |
| | | 1 | | Percentage of free CIs in CA |

*Table 20. ICF Catalog Field Names  (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
|  |  | 1 |  | Percentage of bytes free in CI |
|  |  | 2 |  | Number CIs/ CA |
|  |  | 2 |  | Free bytes/CI |
|  |  | 4 |  | Number of logical records |
|  |  | 4 |  | Number of deleted records |
|  |  | 4 |  | Number of inserted records |
|  |  | 4 |  | Number of updated records |
|  |  | 4 |  | Number of retrieved records |
|  |  | 4 |  | Bytes of free space in component |
|  |  | 4 |  | Number of CI splits |
|  |  | 4 |  | Number of CA splits |
|  |  | 4 |  | Number of EXCPs |
| no | Bitstring | 2 | VSAMTYPE | VSAM data set type information |
|  | First Byte: |  |  |  |
|  | 1... .... |  |  | KSDS=1, not KSDS=0 |
|  | .1.. .... |  |  | Write check |
|  | ..1. .... |  |  | Imbed |
|  | ...1 .... |  |  | Replicate |
|  | .... 1... |  |  | Not defined |
|  | .... .1.. |  |  | Key-range data set |
|  | .... ..1. |  |  | RRDS |
|  | .... ...1 |  |  | Spanned records allowed |
|  | Second Byte: |  |  |  |
|  | 1... .... |  |  | Non-unique or unique keys allowed |
|  | .111 1.1. |  |  | Not defined |
|  | .... .1.. |  |  | LDS |
|  | .... ...1 |  |  | VRRDS |
| no | Bitstring | 2 | VVRNFLGS | Extended format flags |
|  | First Byte: |  |  |  |
|  | 1... .... |  |  | COMUDSIZ and UDATASIZ are invalid |
|  | .111 1111 |  |  | Block level compression |
|  | ..11 1111 |  |  | Not defined |
|  | Second Byte: |  |  |  |
|  | 1111 1111 |  |  | Not defined |
| no | Bitstring | 1 | XACIFLAG | Extended attribute flags |
|  | x... .... |  |  | Reserved, may be on |
|  | .x.. .... |  |  | Data set can be greater than 4GB |
|  | ..xx .... |  |  | Reserved, may be on |
| yes | Fixed | 8 | XHARBA | High-allocated RBA |

*Table 20. ICF Catalog Field Names  (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| no | Fixed | 8 | XHARBADS | Data-set high-allocated RBA |
| yes | Fixed | 8 | XHKRBA | RBA of data control interval with high key |
| no | Fixed | 8 | XHURBADS | Data-set high-used RBA |
| yes | Fixed | 8 | XHURBA | High-used RBA |

**Note:**
> If you attempt to retrieve a 4-byte RBA value (e.g. HARBA, HURBA, HARBADS, HURBADS, or HKRBA) and the value will not fit in the 4-bytes provided, that length of that returned data will be zero as shown under ″Work Area Format Description″ in Appendix D. You can either change to always request the extended fields shown above, or request the setting of XACIFLAG and inspect bit 1 to determine whether or not RBAs can be greater than 4 bytes. If so, then request the fields with the names given here.

## Library Entry Field Names

These names are only valid for tape volume catalogs in DFSMS/MVS.

The REP column refers to fields that can repeat when returned by CSI.

*Table 21. Library Entry Field Names*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| no | Character | 8 | LCBCONID | Library console identification |
| no | Character | 8 | LCBDEVTP | Library device type |
| no | Character | 120 | LCBDESCR | Library Description |
| no | Fixed | 4 | LCBEMPTY | Number of empty slots |
| no | Character | 5 | LCBLIBID | Library Identification |
| no | Flag | 1 | LCBLOGIC | Library logic type |
| no | Fixed | 32 | LCBSCRTH | Number of scratch volumes for all 8 media types |
| no | Fixed | 4 | LCBSLOTS | Number of slots |
| no | Fixed | 32 | LCBTHRES | Library scratch threshold for all 8 media types |

## Volume Entry Field Names

These names are only valid for tape volume catalogs in DFSMS/MVS.

The REP column refers to fields that can repeat when returned by CSI.

*Table 22. Volume Entry Field Names*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
| no | Character | 1 | VCBCHKPT | Volume checkpoint |
| | | | "Y" | Yes |
| | | | "N" | No |
| | | | " " | Unknown |
| no | Character | 10 | VCBCRDT | Volume creation date, YYYY-MM-DD |
| no | Group item | 4 | VCBDEVTP | Volume device type |
| | Fixed | 1 | | Recording Technology<br>- Not defined (0)<br>- 18 tracks (1)<br>- 36 tracks (2)<br>- 128 tracks (3)<br>- 256 tracks (4) |
| | Fixed | 1 | | Media Type<br>- Not defined (0)<br>- Media 1 (1)<br>- Media 2 (2)<br>- Media 3 (3)<br>- Media 4 (4) |
| | Fixed | 1 | | Compact Type<br>- Not defined (0)<br>- IDRC (1) |

*Table 22. Volume Entry Field Names  (continued)*

| Rep | Type | Length | Name | Description |
|-----|------|--------|------|-------------|
|  | Fixed | 1 |  | Special Attribute<br>- Not defined (0)<br>- 18 track/read only (1) |
| no | Character | 10 | VCBEDATE | Volume entry/eject date, YYYY-MM-DD |
| no | Bitstring | 2 | VCBERRST | Volume error status |
| no | Character | 10 | VCBEXPDT | Volume expiration date, YYYY-MM-DD |
| no | Character | 44 | VCBLIBNM | Volume library name |
| no | Character | 1 | VCBLOC | Volume location |
|  |  |  | "L" | Library |
|  |  |  | "S" | Shelf |
|  |  |  | " " | Unknown |
| no | Character | 10 | VCBMOUNT | Volume last mount date, YYYY-MM-DD |
| no | Character | 64 | VCBOWNER | Volume owner information |
| no | Character | 8 | VCBSGRP | Volume storage group |
| no | Character | 32 | VCBSHELF | Volume shelf location |
| no | Flag | 1 | VCBUATTR | Volume user attribute |
|  |  |  | "P" | Private |
|  |  |  | "S" | Scratch |
|  |  |  | " " | Unknown |
| no | Flag | 1 | VCBWPROT | Volume write protection status |
| no | Character | 10 | VCBWRITE | Volume last written date, YYYY-MM-DD |

## Sample Programs

Three sample assembler programs and one sample REXX EXEC come with the IGGCSI00 module. These sample programs are intended to let you try out CSI without having to write your own program. You should be able to get CSI up and running using these programs with only a few JCL changes. The sample programs are delivered "as is" and are not intended to be part of the Licensed Product. You may want to modify them to fit your specific needs.

## IGGCSILC

IGGCSILC is an assembler program that produces output similar to Access Method Services (IDCAMS) LISTCAT NAME. Try running it against an IDCAMS LISTCAT NAMES.

Change the Binder/Linkage Editor SYSLIN DD statement to point to the linkage library for your installation.

The input to IGGCSILC is an 80-byte SYSIN DD record. The catalog name of the catalog to be listed should be left justified in column 1 of the record.

A listing of all entry names in the catalog is printed along with a summary of the number each entry type found and the total of all entries found.

# IGGCSIVG

IGGCSIVG is an assembler program that identifies unused space at the end of a VSAM data set. Basically, it computes the difference of the high-used and the high-allocated relative byte addresses (RBAs) for each VSAM data set in a given catalog. This program is useful in identifying over-allocated space.

Change the Binder/Linkage Editor SYSLIN DD statement to point to the linkage library for your installation. IGGCSIVG will run only with releases of DFSMS.

The input to IGGCSIVG is an 80-byte SYSIN DD record. The catalog name of the catalog to be processed must be left justified and start in column 1 of the input record. Multiple input records can be supplied.

IGGCSIVG prints a summary for each catalog name showing the total unused space by VSAM type. A total for all catalog names supplied is printed after the last catalog name is processed.

# IGGCSIVS

IGGCSIVS is an assembler program that identifies which data sets on a given volume reside in a particular catalog. In the event of a disk drive failure, this program would be useful in identifying which entries in a catalog need to be cleaned up if the data sets were recovered to a different volume serial number.

Change the Binder/Linkage Editor SYSLIN DD statement to point to the linkage library for your installation.

The input to IGGCSIVS is an 80-byte SYSIN DD record. The six- character volume serial number should be left justified and starts in column 1 of the input record. The catalog name of the catalog to be searched starts in column 7. Multiple input records can be supplied.

The output is a listing of all data sets that reside in the given catalog and are on the given volume.

# IGGCSIRX

IGGCSIRX is a REXX EXEC that uses CSI. Move this EXEC to a REXX EXEC library and ensure that the linkage library for IGGCSI00 is accessible by the TSO session. IGGCSIVS will run with any release of DFSMS supported by CSI.

When executed, IGGCSIRX will prompt the user for a filter key. This should be a partially qualified data set name as described for the selection criteria field CSIFILTK. The data set name, its type, and volume serial number(s) are returned to the user's TSO session.

# Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Information Enabling Requests
Dept. DZWA
5600 Cottle Road
San Jose, CA 95193 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

## Programming Interface Information

This publication is intended to help you understand and manage catalogs.

This book documents information NOT intended to be used as a Programming Interface of DFSMS.

This book documents Diagnosis, Modification or Tuning Information, which is intended to help you do diagnosis of DFSMS.

Diagnosis, Modification or Tuning Information is identified where it occurs, by an introductory statement to a chapter or section.

## Trademarks

The following terms are trademarks of the IBM Corporation in the United States, or other countries, or both:

| | |
|---|---|
| DB2 | IBM |
| DFSMS | MVS/ESA |
| DFSMSdfp | OS/390 |
| DFSMSdss | Processor Resource/System Manager |
| DFSMShsm | RACF |
| DFSMSrmm | System/390 |
| DFSMS/MVS | VTAM |
| DFSORT | z/OS |

Other company, product, and service names may be trademarks or service marks of others.

# Glossary

This glossary defines technical terms and abbreviations used in DFSMS documentation. If you do not find the term you are looking for, refer to the index of the appropriate DFSMS manual or view the *IBM Dictionary of Computing Terms* located at:

http://www.ibm.com/networking/nsg/nsgmain.htm

This glossary includes terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The *Information Technology Vocabulary* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published part of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- The *IBM Dictionary of Computing*, New York: McGraw-Hill, 1994.

The following cross-reference is used in this glossary:

**See:** This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.

## A

**access method services.**  A multifunction service program that to manages both VSAM and non-VSAM data sets and integrated catalog facility catalogs or VSAM catalogs. It defines data sets and allocates space for VSAM data sets, VSAM catalogues, and ICF catalogs. It converts indexed-sequential (ISAM) data sets to key-sequenced data sets, modifies data set attributes in the catalog, reorganizes data sets, facilitates data portability among operating systems, creates backup copies of data sets and indexes, helps make inaccessible data sets accessible, lists the records of data sets and catalogs, defines and builds alternate indexes, and converts CVOLs and VSAM catalogs to integrated catalog facility catalogs.

**ACS.**  Automatic class selection.

**active control data set (ACDS).**  A VSAM linear data set that contains an SCDS that has been activated to control the storage management policy for the installation. When activating an SCDS, you determine which ACDS will hold the active configuration (if you have defined more than one ACDS). The ACDS is shared by each system that is using the same SMS configuration to manage storage. See also *source control data set, communications data set*.

**alias entry.**  An entry that relates an alias (alternate entryname) to the real entryname of a user catalog or non-VSAM data set.

**ATL.**  Automated tape library.

**automated tape library data server.**  A device consisting of robotic components, cartridge storage areas, tape subsystems, and controlling hardware and software, together with the set of tape volumes that reside in the library and can be mounted on the library tape drives. Contrast with *manual tape library*. See also *tape library*.

**automatic backup.**  (1) (1) In DFSMShsm, the process of automatically copying data sets from primary storage volumes or migration volumes to backup volumes. (2) (2) In OAM, the process of automatically copying a primary copy of an object from DASD, optical, or tape volume to a backup volume contained in an object backup storage group.

**automatic class selection (ACS).**  A mechanism for assigning SMS classes and storage groups.

**availability.**  For a storage subsystem, the degree to which a data set or object can be accessed when requested by a user.

## B

**backup data set.**  A copy that can be used to replace or reconstruct a damaged data set.

**base configuration.**  The part of an SMS configuration that contains general storage management attributes, such as the default management class, default unit, and default device geometry. It also identifies the systems or system groups that an SMS configuration manages.

**basic catalog structure (BCS).**   The name of the catalog structure in the integrated catalog facility environment. See also *integrated catalog facility catalog*.

# C

**CAS.**   Catalog address space.

**catalog connector.**   A catalog entry, called either a user catalog entry or a catalog connector entry, in the master catalog that points to a user catalog. Contains information on aliases for the user catalog.

**CDSC.**   Cataolog data space cache.

**CFRM.**   Coupling Facility Resource Manager.

**communications data set (COMMDS).**   The primary means of communication among systems governed by a single SMS configuration. The COMMDS is a VSAM linear data set that contains the name of the ACDS and current utilization statistics for each system-managed volume, which helps balance space among systems running SMS. See also *active control data set* and *source control data set*.

**CVOL.**   Control volume.

# D

**DASD volume.**   A DASD space identified by a common label and accessed by a set of related addresses. See also *volume, primary storage, migration level 1, migration level 2*.

**data class.**   A collection of allocation and space attributes, defined by the storage administrator, that are used to create a data set.

**data security.**   Prevention of access to or use of data or programs without authorization. As used in this publication, the safety of data from unauthorized use, theft, or purposeful destruction.

**DFSMS environment.**   An environment that helps automate and centralize the management of storage. This is achieved through a combination of hardware, software, and policies. In the DFSMS environment for MVS, this function is provided by DFSMS, DFSORT, and RACF. See also *system-managed storage*.

**DFSMS.**   See *Data Facility Storage Management Subsystem.*

**DFSMSdfp.**   A DFSMS functional component or base element of z/OS, that provides functions for storage management, data management, program management, device management, and distributed data access.

**DFSMSdss.**   A DFSMS functional component or base element of z/OS, used to copy, move, dump, and restore data sets and volumes.

**DFSMShsm.**   A DFSMS functional component or base element of z/OS, used for backing up and recovering data, and managing space on volumes in the storage hierarchy.

**dictionary.**   A table that associates words, phrases, or data patterns to shorter tokens. The tokens replace the associated words, phrases, or data patterns when a data set is compressed.

**dummy storage group.**   A type of storage group that contains the serial numbers of volumes no longer connected to a system. Dummy storage groups allow existing JCL to function without having to be changed. See also *storage group*.

# E

**ECS.**   Enhanced Catalog Sharing.

**entry.**   A collection of information about a cataloged object in a catalog.

**expiration.**   (1) The process by which data sets or objects are identified for deletion because their expiration date or retention period has passed. On DASD, data sets and objects are deleted. On tape, when all data sets have reached their expiration date, the tape volume is available for reuse. (2) In DFSMSrmm, all volumes have an expiration date or retention period set for them either by vital record specification policy, by user-specified JCL when writing a data set to the volume, or by an installation default. When a volume reaches its expiration date or retention period, it becomes eligible for release.

# F

**filtering.**   The process of selecting data sets based on specified criteria. These criteria consist of fully or partially-qualified data set names or of certain data set characteristics.

# G

**global resource serialization (GRS).**   A component of MVS used for serializing use of system resources and for converting hardware reserves on DASD volumes to data set enqueues.

# H

**hardware configuration definition (HCD).**   An interactive interface in MVS that enables an installation to define hardware configurations from a single point of control.

**HCD.** Hardware Configuration Definition.

# I

**ICFRU.** Integrated Catalog Forward Recovery Utility

**integrated catalog facility catalog.** A catalog that is composed of a basic catalog structure (BCS) and its related volume tables of contents (VTOCs) and VSAM volume data sets (VVDSs). See also *basic catalog structure* and *VSAM volume data set*.

**ISC.** In-storage catalog.

**Interactive Storage Management Facility (ISMF).** The interactive interface of DFSMS/MVS that allows users and storage administrators access to the storage management functions.

**ISMF.** Interactive Storage Management Facitlity.

# M

**Management class.** A named collection of management attributes describing the retention, backup, and class transition characteristics for a group of objects in an object storage hierarchy.

**manual tape library.** Installation-defined set of tape drives defined as a logical unit together with the set of system-managed volumes which can be mounted on the drives.

**migration.** The process of moving unused data to lower cost storage in order to make space for high-availability data. To use the data set, it must be recalled. See also *migration level 1, migration level 2*.

**migration level 1.** DFSMShsm-owned DASD volumes that contain data sets migrated from primary storage volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage, migration level 2*.

**migration level 2.** DFSMShsm-owned tape or DASD volumes that contain data sets migrated from primary storage volumes or from migration level 1 volumes. The data can be compressed. See also *storage hierarchy*. Contrast with *primary storage, migration level 1*.

# N

**non-VSAM entry.** A catalog entry that describes a non-VSAM data set. A non-VSAM entry contains the data set's volume serial number and device type. If the data set resides on a magnetic tape volume, the entry can also identify the data set's file number.

**NVR.** Non-VSAM volume record

# O

**OAM.** Object access method.

**object.** A named byte stream having no specific format or record orientation.

**object access method (OAM).** An access method that provides storage, retrieval, and storage hierarchy management for objects and provides storage and retrieval management for tape volumes contained in system-managed libraries.

**object backup storage group.** A type of storage group that contains optical or tape volumes used for backup copies of objects. See also *storage group*.

**object storage group.** A type of storage group that contains objects on DASD, tape, or optical volumes. See also *storage group*.

**object storage hierarchy.** A hierarchy consisting of objects stored in DB2 table spaces on DASD, on optical or tape volumes that reside in a library, and on optical or tape volumes that reside on a shelf. See also *storage hierarchy*.

**optical volume.** Storage space on an optical disk, identified by a volume label. See also *volume*.

# P

**partitioned data set (PDS).** A data set on direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

**partitioned data set extended (PDSE).** A system-managed data set that contains an indexed directory and members that are similar to the directory and members of partitioned data sets. A PDSE can be used instead of a partitioned data set.

**performance.** (1) (1) A measurement of the amount of work a product can produce with a given amount of resources. (2) (2) In a system-managed storage environment, a measurement of effective data processing speed with respect to objectives set by the storage administrator. Performance is largely determined by throughput, response time, and system availability.

**permanent data set.** A user-named data set that is normally retained for longer than the duration of a job or interactive session. Contrast with *temporary data set*.

**pool storage group.** A type of storage group that contains system-managed DASD volumes. Pool storage groups allow groups of volumes to be managed as a single entity. See also *storage group*.

**primary space allocation.** Amount of space requested by a user for a data set when it is created. Contrast with *secondary space allocation*.

**primary storage.** A DASD volume available to users for data allocation. The volumes in primary storage are called primary volumes. See also *storage hierarchy*. Contrast with *migration level 1, migration level 2*.

# R

**recovery.** The process of rebuilding data after it has been damaged or destroyed, often by using a backup copy of the data or by reapplying transactions recorded in a log.

**Resource Access Control Facility (RACF).** An IBM licensed program that is included in z/OS Security Server and is also available as a separate program for the z/OS and VM environments. RACF provides access control by identifying and verifying the users to the system, authorizing access to protected resources, logging detected unauthorized attempts to enter the system, and logging detected accesses to protected resources.

**Resource Measurement Facility (RMF).** An IBM licensed program or optional element of z/OS, that measures selected areas of system activity and presents the data collected in the format of printed reports, system management facilities (SMF) records, or display reports. Use RMF to evaluate system performance and identify reasons for performance problems.

# S

**secondary space allocation.** Amount of additional space requested by the user for a data set when primary space is full. Contrast with *primary space allocation*.

**shelf location.** (1) A single space on a shelf for storage of removable media. (2) In DFSMSrmm, a shelf location is defined in the removable media library by a rack number, and in a storage location, it is defined by a bin number. See also *rack number, bin number*.

**SMS.** Storage Management Subsystem.

**SMS complex.** A collection of systems or system groups that share a common configuration. All systems in an SMS complex share a common active control data set (ACDS) and a communications data set (COMMDS). The systems or system groups that share the configuration are defined to SMS in the SMS base configuration.

**SMS class.** A list of attributes that the Storage Management Subsystem applies to data sets having

similar allocation (data class), performance (storage class), or availability (management class) needs.

**SMS-managed data set.** A data set that has been assigned a storage class.

**source control data set (SCDS).** A VSAM linear data set containing an SMS configuration. The SMS configuration in an SCDS can be changed and validated using ISMF. See also *active control data set, communications data set*.

**sphere record.** A catalog record which is a collection of logically related records describing a VSAM data set.

**storage administrator.** A person in the data processing center who is responsible for defining, implementing, and maintaining storage management policies.

**storage class.** A collection of storage attributes that identify performance goals and availability requirements, defined by the storage administrator, used to select a device that can meet those goals and requirements.

**storage group.** A collection of storage volumes and attributes, defined by the storage administrator. The collections can be a group of DASD volumes or tape volumes, or a group of DASD, optical, or tape volumes treated as a single object storage hierarchy. See also *VIO storage group, pool storage group, tape storage group, object storage group, object backup storage group, dummy storage group*.

**storage hierarchy.** An arrangement of storage devices with different speeds and capacities. The levels of the storage hierarchy include main storage (memory, DASD cache), primary storage (DASD containing uncompressed data), migration level 1 (DASD containing data in a space-saving format), and migration level 2 (tape cartridges containing data in a space-saving format). See also *primary storage, migration level 1, migration level 2, object storage hierarchy*.

**storage management.** The activities of data set allocation, placement, monitoring, migration, backup, recall, recovery, and deletion. These can be done either manually or by using automated processes. The Storage Management Subsystem automates these processes for you, while optimizing storage resources. See also *Storage Management Subsystem*.

**Storage Management Subsystem (SMS).** A DFSMS facility used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system through data class, storage class, management class, storage group, and ACS routine definitions.

**system data.** The data sets required by z/OS or its subsystems for initialization and control.

**system-managed storage.** Storage managed by the Storage Management Subsystem. SMS attempts to deliver required services for availability, performance, and space to applications. See also *system-managed storage environment*.

**system-managed tape library.** A collection of tape volumes and tape devices, defined in the tape configuration database. A system-managed tape library can be automated or manual. See also *tape library*.

**system management facilities (SMF).** A component of z/OS that collects input/output (I/O) statistics, provided at the data set and storage class levels, which helps you monitor the performance of the direct access storage subsystem.

**system programmer.** A programmer who plans, generates, maintains, extends, and controls the use of an operating system and applications with the aim of improving overall productivity of an installation.

# T

**tape library.** A set of equipment and facilities that support an installation's tape environment. This can include tape storage racks, a set of tape drives, and a set of related tape volumes mounted on those drives. See also *system-managed tape library*.

**Tape Library Dataserver.** A hardware device that maintains the tape inventory that is associated with a set of tape drives. An automated tape library dataserver also manages the mounting, removal, and storage of tapes.

**tape storage group.** A type of storage group that contains system-managed private tape volumes. The tape storage group definition specifies the system-managed tape libraries that can contain tape volumes. See also *storage group*.

**tape volume.** A tape volume is the recording space on a single tape cartridge or reel. See also *volume*.

**temporary data set.** An uncataloged data set whose name begins with & or &&, that is normally used only for the duration of a job or interactive session. Contrast with *permanent data set*.

**threshold.** A storage group attribute that controls the space usage on DASD volumes, as a percentage of occupied tracks versus total tracks. The *low migration threshold* is used during primary space management and interval migration to determine when to stop processing data. The *high allocation threshold* is used to determine candidate volumes for new data set allocations. Volumes with occupancy lower than the high

threshold are selected over volumes that meet or exceed the high threshold value.

# V

**virtual input/output (VIO) storage group.** A type of storage group that allocates data sets to paging storage, which simulates a DASD volume. VIO storage groups do not contain any actual DASD volumes. See also *storage group*.

**VLF.** Virtual lookaside facility

**volume.** The storage space on DASD, tape, or optical devices, which is identified by a volume label. See also *DASD volume, optical volume, tape volume*.

**VSAM.** Virtual storage access method

**VSAM sphere.** The base cluster of a VSAM data set and its associated alternate indexes.

**VSAM volume control record (VVCR).** The first logical record in the VVDS that contains information to manage DASD space and the BCS back pointers.

**VSAM volume data set (VVDS).** A data set that describes the characteristics of VSAM and system-managed data sets residing on a given DASD volume; part of an integrated catalog facility catalog. See also *basic catalog structure, integrated catalog facility catalog*.

**VSAM volume record (VVR).** A VVDS record for VSAM data sets.

# W

**WTOR.** Write-to-operator with reply.

# Index

## A

## B

# Readers' Comments — We'd Like to Hear from You

**z/OS**
**DFSMS: Managing Catalogs**

**Publication No.  SC26-7409-01**

**Overall, how satisfied are you with the information in this book?**

|  | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Overall satisfaction | ☐ | ☐ | ☐ | ☐ | ☐ |

**How satisfied are you that the information in this book is:**

|  | Very Satisfied | Satisfied | Neutral | Dissatisfied | Very Dissatisfied |
|---|---|---|---|---|---|
| Accurate | ☐ | ☐ | ☐ | ☐ | ☐ |
| Complete | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to find | ☐ | ☐ | ☐ | ☐ | ☐ |
| Easy to understand | ☐ | ☐ | ☐ | ☐ | ☐ |
| Well organized | ☐ | ☐ | ☐ | ☐ | ☐ |
| Applicable to your tasks | ☐ | ☐ | ☐ | ☐ | ☐ |

**Please tell us how we can improve this book:**

Thank you for your responses. May we contact you?   ☐ Yes   ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.
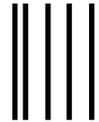
_____
Name

_____
Company or Organization

_____
Phone No.

_____
Address

Fold and Tape                    **Please do not staple**                    Fold and Tape

Fold and Tape                    **Please do not staple**                    Fold and Tape

**IBM** ®

Program Number:  5694-A01